Des codes pour engendrer des langages de mots infinis

Vinh Duc Tran

Année 2011

Thèse de doctorat

présentée à :

L'UNIVERSITÉ NICE – SOPHIA ANTIPOLIS

pour obtenir le titre de Docteur

École Doctorale STIC

spécialité : INFORMATIQUE

par

Tran Vinh Duc

Des codes pour engendrer des langages de mots infinis

Soutenue le 16 décembre 2011 devant la commission d'examen :

Pr.	O. Carton	*Rapporteur*
Pr.	E. Formenti	*Président*
Dr.	S. Julia	*Membre*
Pr.	I. Litovsky	*Directeur*
Pr.	G. Richomme	*Rapporteur*

Remerciements

Ce travail n'aurait jamais vu le jour sans l'aide de Sandrine JULIA. Elle a été l'origine de mon intérêt pour les mots infinis et elle m'a guidé tout au long de ce travail. Qu'elle trouve ici ma profonde gratitude.

Je remercie particulièrement Igor LITOVSKY, mon directeur de thèse, pour tout ce qu'il m'a appris, pour sa disponibilité, et pour ses conseils précieux pendant quatre dernières années.

Je remercie Enrico FORMENTI d'avoir accepté de présider le jury de thèse.

Je remercie Olivier CARTON et Gwénaël RICHOMME pour avoir accepté d'être rapporteurs de ma thèse. Je les remercie aussi de ses remarques, suggestions, et critiques.

Je remercie Gabriel FICI d'avoir accepté de faire partie de mon jury.

Merci à tous mes amis en France, en particulier Chi-Anh, Jérôme, Vinh, Tram et Tuan qui m'ont beaucoup aidé. Grâce à eux, mon séjour en France devient beaucoup plus agréable.

Enfin, je remercie ma famille qui m'a toujours soutenu tout au long de mes études.

Table des matières

Introduction

Dans cette thèse on s'intéresse aux langages L de mots infinis qui sont la puissance infinie (puissance ω) de langages finis de mots finis. Si R est un tel langage, on note R^ω sa puissance infinie, et on a donc $L = R^\omega$. Cette puissance ω est donc une application de l'ensemble des langages de mots dans l'ensemble des langages de mots infinis et elle est définie comme étant l'ensemble de toutes les concaténations d'une infinité de mots de R. Ainsi à toute suite de mots de R est associée par concaténation des mots de cette suite, un mot infini de L et on se pose la question suivante : étant donné un langage fini R, existe-t-il un langage C fini ou pas, tel que d'une part $R^\omega = C^\omega$ et d'autre part tel que l'application qui à toute suite de mots de C associe un mot infini, soit injective ? De tels langages C sont appelés ω-code [Sta86], en effet ce sont des codes par rapport à l'opération puissance ω et bien sûr tout ω-code est un code, mais la réciproque est fausse.

Cette question « assez naturelle et classique », par exemple si on considère la concaténation finie (et non infinie) il s'agit de savoir si un monoïde est ou non libre, n'a été étudiée que par peu de personnes, et elle reste actuellement toujours non résolue.

Un des points de départ de ce travail a été le suivant : on sait [JLP96] que si R est un code tel que R^+ est le plus grand générateur de R^ω, alors il existe un ω-code C tel $R^\omega = C^\omega$ si et seulement si R lui-même est un ω-code. Ainsi, on a essayé de définir une classe de langage qui serait la plus « simple » possible après la classe des codes. Les codes sont les langages C pour lesquels deux concaténations différentes de mots de C, donnent deux résultats différents, donc il y a 0 égalité entre deux concaténations différentes de mots de C. La classe la plus simple après les codes pourrait donc être la classe des langages R ayant « une seule » égalité entre deux concaténations différentes

de mots de R. Mais dès qu'il y a une égalité, une infinité d'autres en découlent, aussi on définira la classe des langages R tels qu'il existe une unique égalité e entre deux concaténations différentes de mots de R, telle que toutes les autres égalités entre deux concaténations différentes de mots de R sont exactement toutes les égalités qui découlent de e. De tels langages sont appelés langages à un relateur. Pour cette vaste classe de langages, on caractérise les langages finis R tels qu'il existe un ω-code ou un code C tels que $R^\omega = C^\omega$ et on montre que C n'est jamais un langage fini.

D'après [DLLS94], les codes peuvent être caractérisés par les factorisations de mots infinis : un langage C est un code si et seulement si pour tout mot $u \in C^+$, le mot infini u^ω se décompose de façon unique en mots de C. Cela nous a amenés à définir la classe des langages n-réduits. Un langage L est n-réduit si pour tout mot $u \in L^k$, $k \leq n$, le mot infini u^ω se décompose de façon unique en mots de L. Par définition, un langage $n+1$-réduit est n-réduit, mais la réciproque est fausse ; et un code est un langage ∞-réduit. Nous considérons les propriétés de ces langages en tant que générateurs de langages de mots infinis.

Après l'introduction, la thèse est divisée en cinq chapitres :

Dans le chapitre 1, nous présentons des notations et des résultats sur les langages de mots infinis et les codes, résultats que nous utiliserons dans les chapitres ultérieurs.

Le chapitre 2 est un exposé rapide des résultats concernant les générateurs de langages de mots infinis et la question de savoir s'il existe un code ou un ω-code générateur pour un langage de mots infinis.

Dans le chapitre 3, nous étudions l'ensemble des relateurs satisfaits par un langage L, c'est-à-dire les couples (u, v) où u et v sont deux factorisations d'un même mot dans $L^* \cup L^\omega$. Avec cette notation, nous établissons des conditions nécessaires pour qu'une ω-puissance ait un code ou un ω-code générateur. Nous montrons aussi la méthode pour calculer l'ensemble des relateurs d'un langage fini.

Dans le chapitre 4, nous définissons la classe de langages à un relateur et présentons le résultat principal de thèse, qui nous permet de caractériser les langages à un relateur L tel que L^ω a un code ou un ω-code générateur.

Dans le chapitre 5, nous définissons les langages n-réduits. Nous dégageons des propriétés de ces langages. Nous illustrons aussi l'existence des langages de mots infinis

qui n'ont pas de générateur réduit.

Chapitre 1

Préliminaires

Dans ce chapitre, nous présentons des notations et des résultats sur les langages de mots infinis [PP02, Tho90, Eil74] et les codes [BPR09, Sta86], résultats que nous utiliserons dans les chapitres ultérieurs.

1.1 Mots et mots infinis

Soit A un ensemble nommé *alphabet*. Les éléments de cet ensemble sont appelés *lettres*. Un *mot fini* sur l'alphabet A est une suite finie de lettres de A. Le nombre n de lettres du mot u est la *longueur* du mot u et elle est notée $|u|$. On note $|u|_a$ le nombre d'occurrences de la lettre a dans u. La suite vide est notée ε et elle est appelée *mot vide*. On note A^* l'ensemble des mots finis sur l'alphabet A et on note $A^+ = A^* - \varepsilon$.

Un *mot infini* sur l'alphabet A est une suite infinie de lettres de A. On note A^ω l'ensemble des mots infinis sur l'alphabet A et on pose :

$$A^\infty = A^* \cup A^\omega.$$

On munit A^∞ d'une structure de monoïde prolongeant celle de A^* en posant :

$$uv = \begin{cases} uv & \text{si} \quad u \in A^* \\ u & \text{si} \quad u \in A^\omega. \end{cases}$$

Un mot $w \in A^*$ est *facteur* du mot $x \in A^\infty$ s'il existe $u \in A^*$ et $v \in A^\infty$ tels que $x = uwv$; le mot w est *préfixe* de x, noté $w \leq x$, si $u = \varepsilon$; le mot w est *préfixe propre* de x, noté $w < x$, si $u = \varepsilon$ et $v \neq \varepsilon$. Un mot $w \in A^\infty$ est *suffixe* du mot $x \in A^\infty$ s'il existe $u \in A^*$ tel que $x = uw$. On note respectivement $\mathrm{Fact}(x), \mathrm{Pref}(x)$ et $\mathrm{Suff}(x)$ l'ensemble des facteurs, préfixes et suffixes du mot x. Etant donnée une partie X de A^∞, on note respectivement $\mathrm{Fact}(X), \mathrm{Pref}(X)$ et $\mathrm{Suff}(X)$ l'ensemble des facteurs, des préfixes et des suffixes des mots de X.

Remarque 1.1.1. Si $X \subseteq A^\omega$ alors $\mathrm{Pref}(X) \subseteq A^*, \mathrm{Fact}(X) \subseteq A^*$ et $\mathrm{Suff}(X) \subseteq A^\omega$.

Un mot infini $u = a_0 a_1 \ldots$ est dit *ultimement périodique* s'il existe deux entiers m, n tels que $a_i = a_{i+n}$ pour tout $i \geq m$; ce mot u est *périodique* si m peut être choisi égal à 0.

Un mot $z \in A^+$ est *primitif* si $z = u^n$ seulement pour $n = 1$, *imprimitif* dans le cas contraire. Tout mot $z \in A^+$ possède une unique racine notée $\sqrt{z}$ qui est le mot primitif vérifiant $(\sqrt{z})^n = z$ pour un entier n.

Les deux théorèmes suivants sont connus, voir [Lot02].

Théorème 1.1.2. *Deux mots $u, v \in A^+$ commutent, c'est-à-dire $uv = vu$, si et seulement s'ils ont la même racine primitive.*

Théorème 1.1.3. *Soient $x \in A^+$ et $y, z \in A^*$. Si $xz = yx$, alors il existe deux mots α, β et un entier positif k tels que $x = (\alpha\beta)^k\alpha$, $y = \alpha\beta$ et $z = \beta\alpha$.*

1.2 Langages rationnels

Les parties de A^* sont appelées des *langages*, celles de A^ω des ω-langages, celles de A^∞ des ∞-langages. Pour deux langages X et Y donnés, on définit les opérations *rationnelles* :

(1) l'union $X \cup Y$,

(2) le produit $XY = \{xy : x \in X \text{ et } y \in Y\}$,

(3) l'étoile $X^* = \{x_1 \ldots x_n : n \geq 0 \text{ et } x_1, \ldots, x_n \in X\}$.

Donc X^* est le sous-monoïde de A^* engendré par X.

On définit aussi l'opération $^+$ par

$$X^+ = \{x_1 \dots x_n : n > 0 \text{ et } x_1, \dots, x_n \in X\}$$

qui est le sous-semigroupe engendré par X.

L'ensemble des *langages rationnels* de A^* est la plus petite famille $\mathrm{Rat}(A^*)$ de sous-ensembles de A^* telle que :

(a) $\mathrm{Rat}(A^*)$ contient l'ensemble vide et les singletons $\{a\}$ pour tout $a \in A$;

(b) $\mathrm{Rat}(A^*)$ est fermé par union finie, produit fini et étoile.

Remarque 1.2.1. Le langage $\{\varepsilon\}$ est rationnel car $\emptyset^* = \{\varepsilon\}$.

On étend de façon naturelle le produit de deux langages en posant pour $X \subseteq A^*$ et $Y \in A^\infty$,

$$XY = \{xy : x \in X \text{ et } y \in Y\}.$$

On définit également la puissance infinie (ou l'ω-puissance) d'un langage $X \subseteq A^*$ par

$$X^\omega = \{x_0 x_1 \cdots : \text{ pour tout } i \geq 0,\ x_i \in X - \varepsilon\}.$$

On pose

$$X^\infty = X^* \cup X^\omega.$$

Remarque 1.2.2. Par définition, pour tout langage X, on a

$$X^\omega = (X - \varepsilon)^\omega.$$

Donc $\emptyset^\omega = \{\varepsilon\}^\omega = \emptyset$.

Exemple 1. Soit $A = \{a, b\}$ un alphabet. L'ensemble $L = a^*b$ est un langage sur A. Son ω-puissance L^ω est l'ω-langage qui contient tous les ω-mots ayant une infinité de fois la lettre b.

L'ensemble des *langage rationnels* de A^∞ est la plus petite famille $\mathrm{Rat}(A^\infty)$ de sous-ensembles de A^∞ telle que :

(a) $\mathrm{Rat}(A^\infty)$ contient l'ensemble vide et les singletons $\{a\}$ pour tout $a \in A$,

(b) $\mathrm{Rat}(A^\infty)$ est fermé par union finie, produit fini, étoile et ω-puissance.

Le théorème suivant [PP02] caractérise les langages rationnels de A^ω.

Théorème 1.2.3. *Un ω-langage est un langage rationnel de A^ω si et seulement s'il est union finie des langages de la forme XY^ω où X et Y sont des langages rationnels de A^*.*

1.3 Limite et adhérence

Soit $L \subseteq A^*$ un langage, la *limite* de L, notée $\overrightarrow{L}$, est l'ensemble des ω-mots ayant une infinité de préfixes dans L :

$$\overrightarrow{L} = \{w \in A^\omega : Card(\mathrm{Pref}(w) \cap L) \text{ est infini}\}.$$

Il existe la relation suivante entre la puissance ω et la limite [Sta80] :

$$\overrightarrow{L^*} = L^\omega \cup L^* \overrightarrow{L}$$

On dit que l'ω-langage K est *rationnel déterministe* s'il existe un langage rationnel de mots finis $L \subseteq A^*$ tel que $K = \overrightarrow{L}$.

Pour tout langage L, son *adhérence* est un ω-langage défini par

$$\begin{aligned} Adh(L) &= \{w \in A^\omega : \mathrm{Pref}(w) \subseteq \mathrm{Pref}(L)\} \\ &= \overrightarrow{\mathrm{Pref}(L)}. \end{aligned}$$

On a la relation suivante entre la puissance $^\omega$ et l'adhérence [BN80] :

$$Adh(L^*) = L^\omega \cup L^* Adh(L)$$

En particulier, quand L est un langage fini, on a :

$$Adh(L) = \overrightarrow{L} = \emptyset \qquad \text{et} \qquad \overrightarrow{L^*} = Adh(L^*) = L^\omega.$$

On dit que l'ω-langage K est une *adhérence* s'il existe un langage $L \subseteq A^*$ tel que $K = Adh(L)$.

Exemple 2. Soit $A = \{a, b\}$ et soit $L = b^+a^+$ un langage sur A. L'ω-puissance $L^\omega = (b^+a^+)^\omega$ est l'ensemble des ω-mots ayant une infinité de a, une infinité de b et commençant par b. Sa limite $\overrightarrow{L} = b^+a^\omega$ tandis que son adhérence est $Adh(L) = b^\omega \cup b^+a^\omega$. De plus, L^ω est strictement inclus dans $\overrightarrow{L^*} = bA^\omega - A^*b^\omega$, par exemple $ba^\omega \notin L^\omega$; et $\overrightarrow{L^*}$ est strictement inclus dans $Adh(L^*) = bA^\omega$.

Les trois opérations, la puissance ω, la limite et l'adhérence nous permettent de construire des ω-langages à partir d'un langage finitaire.

1.4 Codes et mots infinis

Dans cette section, on considère une classe particulière de langages : celle des codes [BPR09]. Ces langages assurent l'unicité du décodage des mots.

Soit $C \subseteq A^+$ un langage. Une *factorisation* sur C (ou C-factorisation) d'un mot $u \in A^+$ est une suite finie de mots de C : $(x_1, x_2, \ldots, x_n)$, $n \geq 1$ telle que $u = x_1x_2 \ldots x_n$. Une factorisation sur C (ou C-factorisation) d'un ω-mot $w \in A^\omega$ est une suite infinie de C : $(x_1, x_2, \ldots)$ telle que $w = x_1x_2 \ldots$.

Proposition 1.4.1. *[BPR09] Soit $C \subseteq A^+$. Les assertions suivantes sont équivalentes :*

(i) Tout mot de C^+ a une seule C-factorisation.

(ii) Pour tous $u, v \in C$, on a :

$$uC^* \cap vC^* \neq \emptyset \quad \textit{implique} \quad u = v.$$

(iii) $C^{-1}C \cap C^*(C^*)^{-1} = \{\varepsilon\}$.

Un langage $C \subseteq A^+$ vérifiant les conditions *(i)–(iii)* ci-dessus est appelé *code*.

Les codes nous permettent de décoder de façon unique les mots finis mais ce n'est pas le cas en général pour les mots infinis. Les ω-codes qui sont définis par L. Staiger

dans [Sta86] assurent l'unicité du décodage des mots infinis. En dépit de leur nom, les ω-codes sont des langages de mots finis.

Proposition 1.4.2. *[Sta86] Soit $C \subseteq A^+$. Les assertions suivantes sont équivalentes :*

(i') *Tout mot de C^ω a une seule C-factorisation.*

(ii') *Pour tous $u, v \in C$, on a :*

$$uC^\omega \cap vC^\omega \neq \emptyset \quad \textit{implique} \quad u = v.$$

(iii') $C^{-1}C \cap C^\omega(C^\omega)^{-1} = \{\varepsilon\}$.

Un langage $C \subseteq A^+$ vérifiant les conditions *(i')–(iii')* ci-dessus est appelé *ω-code*.

Remarque 1.4.3. Par définition d'un ω-code, on a les propriétés suivantes :

- L'ensemble $\emptyset$ est un ω-code.
- Un singleton $\{u\}$ est un ω-code si et seulement si $u \neq \varepsilon$.
- Tout sous-ensemble d'un ω-code est un ω-code.
- Les ω-codes sont des codes et il existe des codes non ω-codes.

Exemple 3. Le langage $C = \{a, ab, b^2\}$ est un code mais ce n'est pas un ω-code car le mot $ab^\omega \in C^\omega$ a deux C-factorisations : $(ab, bb, bb, \dots)$ et $(a, bb, bb, \dots)$.

Lemme 1.4.4. *[Jul97] Soit C un langage. Tout ω-mot ultimement périodique de C^ω admet une C-factorisation ultimement périodique.*

Lemme 1.4.5. *Soient $x, y, z \in A^+$. On a $y^\omega = xz^\omega$ si et seulement s'il existe $i, j > 0$ tels que $y^i x = xz^j$.*

Démonstration. Si $y^\omega = xz^\omega$ alors il existe un entier $n > 0$ et un mot $t \in A^*$ tels que $y^n = xt$. Donc on a $y^\omega = (xt)^\omega = xz^\omega$. Alors $(tx)^\omega = z^\omega$. Donc il existe k, j tels que $(tx)^k = z^j$. On a

$$x(tx)^k = xz^j \quad \text{et} \quad xt = y^n$$

Posons $i = nk$, on a $y^i x = xz^j$.

Réciproquement, s'il existe $i, j > 0$ tels que $y^i x = xz^j$, alors on a $y^{ni}x = xz^{nj}$ pour chaque $n > 0$. Donc les deux mots y^ω et xz^ω ont une infinité de préfixes communs. On en déduit que $y^\omega = xz^\omega$. □

On pourra trouver dans [BPR09] des caractérisations des codes concernant les mots finis. Cependant, les codes peuvent aussi être caractérisés par le fait de factoriser les mots infinis [DLLS94].

Proposition 1.4.6. *Soit $C \subseteq A^+$. Les assertions suivantes sont équivalentes :*

(i) C est un code.

(ii) Pour chaque $u \in C^+$, u^ω a une unique C-factorisation.

Démonstration. $\neg(i) \Rightarrow \neg(ii)$: Supposons que C n'est pas un code, alors il existe $y \in C^+$ tel que y a deux C-factorisations. Donc y^ω a aussi deux C-factorisations.

$\neg(ii) \Rightarrow \neg(i)$: Supposons qu'il existe $y = y_1 \dots y_n \in C^+$ tel que y^ω a deux C-factorisations de premiers pas distincts : $(y_1 \dots y_n)^\omega = (x_1 x_2 \dots)$ avec $x_1 \neq y_1$. D'après le lemme 1.4.4, il existe $t, z \in C^+$ tel que $x_2 x_3 \cdots = t z^\omega$. Donc on a

$$y^\omega = x_1 t z^\omega$$

D'après le lemme 1.4.5, il existe $i, j > 0$ tels que

$$y^i x_1 t = x_1 t z^j.$$

Alors C n'est pas un code. □

Les plus simples des codes sont les codes préfixes. Rappelons qu'un langage non vide est un *code préfixe* s'il ne contient pas deux mots dont l'un est préfixe propre de l'autre. De façon symétrique, un langage est un *code suffixe* s'il ne contient pas deux mots dont l'un est suffixe propre de l'autre. Un langage est un *code bifixe* s'il est préfixe et suffixe.

En général, le décodage d'un message ne peut être réalisé que si on reçoit le message entier. Naturellement, nous nous intéressons aux codes qui nous permettent de décoder le message en lisant à partir du début vers la droite avec quelques pas en différé. Ce sont les codes à délai borné [GM59, Sta86].

Un langage C est un *code à délai borné* s'il existe un entier $m \geq 0$ pour lequel la

propriété suivante est vérifiée : pour tous $u, v \in C$,

$$uC^m A^\omega \cap vC^\omega \neq \emptyset \quad \text{implique} \quad u = v. \tag{1.1}$$

Le nombre m qui intervient dans la définition précédente est appelé *délai* de C.

Un langage qui vérifie la condition (1.1) est nécessairement un code. En effet, on verra que la classe des codes à délai borné est une sous-classe de celle des ω-codes. De plus, la classe des codes à délai borné 0 et celle des codes préfixes coïncident.

Exemple 4. Le code préfixe $P = \{a, ba\}$ est donc un code à délai borné 0. Le code $C = \{a, a^d b\}$ est un code à délai borné d. Le code suffixe $C = \{a, ab, b^2\}$ est sans délai borné car pour tout $m \geq 0$, le mot $a(b^2)^m$ est un préfixe de $ab(b^2)^m b^\omega$.

La proposition suivante [DLLS94] nous donne une caractérisation des codes à délai borné.

Proposition 1.4.7. *Un code C est à délai borné si et seulement si C est un ω-code et $C^\omega \cap C^* Adh(C) = \emptyset$.*

De plus, par définition, nous savons que l'adhérence d'un langage fini est vide. Ainsi on a

Corollaire 1.4.8. *Un ω-code fini est un code à délai borné.*

Exemple 5. L'ω-code $C = \{a, b\} \cup ab^*c$ n'est pas un code à délai borné. On s'aperçoit qu'effectivement $ab^\omega \in C^\omega \cap C^* Adh(C)$.

Remarque 1.4.9. On a les relations suivantes entre les différentes classes de codes :

$$\text{codes préfixes} \subsetneq \text{codes à délai borné} \subsetneq \omega\text{-codes} \subsetneq \text{codes}$$

La notion de délai s'applique à des langages quelconques, même si ce ne sont pas des codes. Un mot infini peut avoir plusieurs factorisations sur un langage à délai borné L, mais si on lit quelques mots, alors le premier de ces mots est de façon sûre le premier mot dans une des factorisations sur L. En comparant avec les codes à délai borné, pour les langages à délai, la condition de l'unicité des factorisations est retirée, mais la condition du délai sur la construction des factorisations est maintenue.

Un langage $L \subseteq A^+$ est appelé *langage à délai borné* s'il existe un entier $m \geq 0$ pour lequel la propriété suivante est vérifiée : pour tout $u \in L$,

$$uL^m A^\omega \cap L^\omega \subseteq uL^\omega. \tag{1.2}$$

Dans ce cas, L est dit à délai m. Le délai de L est le plus petit entier m vérifiant la propriété (1.2).

Exemple 6. Le délai du langage $\{a^2, a^3\}$ est 0. Quant au délai du langage $\{a^2, a^3, b\}$, c'est 1.

La proposition suivante nous donne une relation entre les codes et les langages à délai borné.

Proposition 1.4.10. *[DLLS94] Un code qui vérifie la propriété* (1.2) *est un code à délai borné.*

Chapitre 2

Codes générateurs de langages de mots infinis

Ce chapitre est un exposé rapide des résultats concernant les générateurs de langages de mots infinis et la question de savoir s'il existe un code ou un ω-code générateur pour un langage de mots infinis.

2.1 Générateurs de langages de mots infinis

Un ω-langage X est dit être une ω-*puissance* si $X = L^\omega$ pour un langage $L \subseteq A^+$. Le langage L est appelé *générateur* de X. On note $[X]$ la famille des générateurs d'un ω-langage X, c'est-à-dire :

$$[X] = \{L \subseteq A^+ : L^\omega = X\}.$$

Exemple 1 (reprise de p. 3). Soit $A = \{a, b\}$ un alphabet et soit X l'ω-langage sur A qui contient tous les ω-mots ayant une infinité de fois la lettre b. C'est-à-dire :

$$X = \{w \in A^\omega : |w|_b = \infty\} = (a^*b)^\omega.$$

Donc X est une ω-puissance et a^*b est un générateur de X. Un autre générateur de X est A^*bA^*.

Exemple 7. Soit $A = \{a, b\}$ un alphabet et soit X l'ω-langage sur A qui contient tous les ω-mots ayant un nombre fini de fois la lettre b. C'est-à-dire :

$$X = \{w \in A^\omega : |w|_b < \infty\} = \{a, b\}^* a^\omega.$$

Supposons qu'il existe un langage G tel que $G^\omega = X$. Comme $ba^\omega \in X$, on a $ba^i \in G$ pour un entier $i \geq 0$. Donc $(ba^i)^\omega \in G^\omega$ mais $(ba^i)^\omega \notin X$. C'est une contradiction. Donc $[X] = \emptyset$.

La proposition suivante [LT87] porte sur la maximalité des générateurs d'ω-langages rationnels.

Proposition 2.1.1. *Soit X un ω-langage rationnel. On peut décider si la famille $[X]$ des générateurs est vide. Si $[X] \neq \emptyset$, alors $[X]$ admet un nombre fini non nul de générateurs maximaux (pour l'inclusion). Ces générateurs maximaux sont des semi-groupes rationnels et ils peuvent être construits à partir d'un automate reconnaissant X. De plus, tout générateur de X est inclus dans un des générateurs maximaux.*

Exemple 8. Soit $A = \{a, b\}$ et soit $L = A^*\{a^2, b^2\}A^*$. L'ω-puissance L^ω a deux générateurs maximaux M_1 et M_2 où

$$M_1 = L \cup (ab)^* a \quad \text{et} \quad M_2 = L \cup (ba)^* b.$$

Dans le cas où l'ω-langage X a un unique générateur maximal, alors ce générateur est le plus grand générateur. On a donc [LT87] :

Corollaire 2.1.2. *On peut décider si un ω-langage rationnel admet un plus grand générateur.*

Si $L \subseteq A^+$ est un générateur de l'ω-langage $X \neq \emptyset$, les langages $G = L - L^+L$ et G^2 sont aussi générateurs de X. De plus $G \cap G^2 = \emptyset$. Alors on a :

Proposition 2.1.3. *Un ω-langage non vide n'a jamais de plus petit générateur.*

Dans l'étude de la famille des générateurs $[X]$ de l'ω-langage X, les deux majorants de $[X]$ suivants ont un rôle important dans la compréhension de la famille $[X]$. Le premier, c'est le stabilisateur de X, noté $Stab(X)$, qui est défini par

$$Stab(X) = \{u \in A^+ : uX \subseteq X\}.$$

Le second, c'est le langage caractéristique de X, qui est défini par

$$\chi(X) = \{u \in Stab(X) : u^\omega \in X\}.$$

Proposition 2.1.4. *[Lit88] Soit L un langage, $Stab(L^\omega)$ et $\chi(L^\omega)$ sont deux majorants de $[L^\omega]$, c'est-à-dire, pour tout $G \in [L^\omega]$, $G \subseteq \chi(L^\omega) \subseteq Stab(L^\omega)$.*

La propriété suivante [Lit89] porte sur la constructibilité des langages $Stab$ et χ.

Proposition 2.1.5. *Soit X un ω-langage rationnel, $Stab(X)$ et $\chi(X)$ sont rationnels et ils peuvent être construits (à partir d'un automate reconnaissant X).*

Les langages $Stab(X)$ et $\chi(X)$ peuvent être ou non générateurs de L^ω comme l'indique l'exemple suivant.

Exemple 9. (a) Soit $X_1 = a^\omega$. Alors $Stab(X_1) = \chi(X_1) = a^+$ est un générateur de X_1.

(b) Soit $X_2 = (a^*b)^\omega$. Le langage $Stab(X_2) = \{a, b\}^+$ n'est pas générateur de X_2 car $a^\omega \notin X_2$. Mais le langage $\chi(X_2) = \{a, b\}^*b\{a, b\}^*$ est un générateur de X_2.

(c) Soit $X_3 = (A^*\{a^2, b^2\})^\omega$. On a $\{a, b\} \subseteq \chi(X_3)$ mais $(ab)^\omega \notin X_3$. Donc $\chi(X_3)$ n'est pas un générateur de X_3.

Proposition 2.1.6. *[Lit88] Soit L un langage rationnel. Si L^ω est une adhérence, alors L^ω admet un plus grand générateur qui est $Stab(L^\omega)$.*

Le langage χ nous donne une caractérisation d'une ω-puissance rationnelle, c'est-à-dire : deux ω-puissances rationnelles sont égales si elles ont même ensemble χ, et réciproquement, d'où la proposition suivante [Jul97] :

Proposition 2.1.7. *Soient L^ω et G^ω deux ω-puissances rationnelles, $L^\omega = G^\omega$ si et seulement si $\chi(L^\omega) = \chi(G^\omega)$.*

On dit qu'une ω-puissance rationnelle X est *finiment engendrée* s'il existe un langage fini F tel que $F^\omega = X$.

Proposition 2.1.8. *[Lit89] Si une ω-puissance rationnelle L^ω est finiment engendrée, alors L^ω est une adhérence. Donc $Stab(L^\omega)$ est le plus grand générateur de L^ω.*

Proposition 2.1.9. *[LT86] On peut décider si une ω-puissance rationnelle est finiment engendrée.*

2.2 Codes et générateurs

Nous nous intéressons au problème ouvert de l'existence d'ω-codes générateurs de langages de mots infinis (voir [Bru91] et aussi [Lot02] p. 229).

Problème 1. *Soit L^ω une ω-puissance rationnelle, décider s'il existe un ω-code C tel que $L^\omega = C^\omega$.*

Avant de discuter en détails de ce problème, nous considérons les problèmes similaires pour les mots finis : celui de l'existence de codes générateurs d'un monoïde.

Soit M un sous-monoïde de A^*, c'est-à-dire $M^* = M$, l'ensemble des générateurs de M admet toujours un plus grand élément (pour l'inclusion), c'est M lui-même. De plus, M a un plus petit générateur, c'est sa racine :

$$Rac(M) = (M - \varepsilon) - (M - \varepsilon)^2.$$

Proposition 2.2.1. *[BPR09] Si M est un sous-monoïde libre de A^*, alors son plus petit générateur est un code. Réciproquement, si $C \subseteq A^+$ est un code, alors le sous-monoïde C^* est libre et C est le plus petit générateur de C^*.*

Donc le problème de l'existence des codes générateurs d'un monoïde M se ramène au problème de tester si le langage $Rac(M)$ est un code ; ce problème est décidable [SP53] dans le cas rationnel.

La situation est plus compliquée pour les ω-puissances : la famille des générateurs ne possède pas de plus petit générateur et elle n'admet pas toujours de plus grand générateur. Dans le cas où le plus grand générateur M existe (par exemple si M^ω est finiment engendré), il se peut que $Rac(M)$ ne soit pas un ω-code mais que M^ω ait un ω-code générateur comme l'indique l'exemple suivant.

Exemple 10. Soit $L = \{a^2, a^3, b\}$. L est la racine du plus grand générateur de L^ω mais L n'est pas un ω-code. Cependant, le langage $\{a^2, a^3b, b\}$ est un ω-code générateur de L^ω.

Dans le cas où le plus grand générateur existe et qu'il est libre, on sait décider du problème 1.

Théorème 2.2.2. *[JLP96] Soit L^ω une ω-puissance rationnelle telle que son plus grand générateur M est un semi-groupe libre, c'est-à-dire $M = C^+$ pour un code C. Alors L^ω a un ω-code générateur si et seulement si C est lui-même un ω-code.*

Si on remplace ω-code par code préfixe, le problème 1 a été résolu par I. Litovsky dans [Lit91].

Théorème 2.2.3. *On peut décider si un ω-langage rationnel X admet un code préfixe générateur, c'est-à-dire si $X = P^\omega$ pour un code préfixe P.*

Nous présentons ici une condition nécessaire pour qu'un ω-langage soit engendré par un code à délai borné [Sta86].

Proposition 2.2.4. *Soit X un ω-langage rationnel. Si X a un code à délai borné générateur, alors X est un langage rationnel déterministe.*

Cette condition n'est pas suffisante comme l'indique l'exemple suivant :

Exemple 11. L'ω-langage $X = \{a, ab, b^2\}^\omega$ est une adhérence car il est finiment engendré, donc il est déterministe. De plus, X admet le plus grand générateur $\{a, ab, b^2\}^+$ qui est un semi-groupe libre, et $\{a, ab, b^2\}$ n'est pas un ω-code. D'après le théorème 2.2.2, X n'a pas d'ω-code générateur. Donc X n'est pas engendré par un code à délai.

L'exemple suivant dans [Sta86] montre qu'il existe un ω-code C tel que C^ω n'est pas déterministe.

Exemple 12. Le langage $C = \{a, ba\} \cup \{ab, ac\}^* aca$ est un ω-code mais C^ω n'est pas rationnel déterministe.

Le problème 1 peut se formuler pour la classe des codes.

Problème 2. *Soit L^ω une ω-puissance rationnelle, décider s'il existe un code C tel que $L^\omega = C^\omega$.*

L'exemple suivant [JT07] montre l'existence d'ω-puissances rationnelles qui n'ont pas de code générateur.

Exemple 13. Soit $A = \{a, b\}$ et soit $L = \{a^2, a^3, b, ba\}$. On a $L^\omega = \{a^2, b\}A^\omega$. Supposons que C soit un code générateur de L^ω. Montrons d'abord que : pour tout $u \in A^*$ et pour tout entier $i > 0$, on a

$$\{au, aua^i\} \not\subseteq C. \tag{2.1}$$

En effet, si $au \in C$, alors $(au)^\omega \in a^2A^\omega$; ainsi pour tout entier $i > 0$, on a $(a^iau)^\omega \in a^2A^\omega \subseteq C^\omega$; puis l'ω-mot $(aua^i)^\omega$ a deux factorisations sur $C \cup \{aua^i\}$:

$$(aua^i)^\omega = au(a^iau)^\omega.$$

D'après la proposition 1.4.6, on a $aua^i \notin C$ ce qui achève la preuve de la relation (2.1).

Comme $a^\omega \in C^\omega$ et par (2.1), il existe un unique entier $i_0 \geq 0$ tel que $aa^{i_0} \in C$. Ensuite, comme $aba^\omega \notin C^\omega$ et par (2.1), l'ω-mot $aa^{i_0}aba^\omega \in C^\omega$ n'a pas de C-factorisation qui commence par un préfixe de $aa^{i_0}a$, donc il existe un unique entier $i_1 \geq 0$ tel que $a^{i_0}aba^{i_1} \in C$. Et ainsi de suite, on définit de façon unique la suite infinie $(i_j)_{j\geq 0}$.

Maintenant, on considère l'ω-mot :

$$w = aa^{i_0}aba^{i_1}ab^{i_2} \cdots \in L^\omega.$$

Il n'a pas de C-factorisation, donc C n'est pas un générateur de L^ω. Alors L^ω n'a pas de code générateur.

Dans le cas où la racine du plus grand générateur d'une ω-puissance est un langage à délai, le résultat suivant [Jul97] réduit les problèmes 1 et 2 à celui de l'existence

d'un ω-code *fini* générateur.

Proposition 2.2.5. *Soit L un langage à délai borné et tel que L^+ est le plus grand générateur de L^ω. Si L^ω est une adhérence alors chaque code C tel que $C^\omega = L^\omega$ est un ω-code fini.*

2.3 Exemples

Dans le cas où le langage $L \neq \{\varepsilon\}$ est singleton, L est un ω-code. Donc L est un ω-code générateur de L^ω. On considère le cas des langages à deux éléments.

Le lemme suivant est bien connu (voir par exemple [Dev93]).

Lemme 2.3.1. *Un langage à deux éléments $\{u, v\} \subseteq A^+$ n'est pas un ω-code si et seulement si u et v commutent.*

Proposition 2.3.2. *Soit $L \subset A^+$ un langage à deux éléments. Alors L^ω admet un ω-code générateur.*

Démonstration. Si $L = \{u, v\} \subset A^+$ est un ω-code, alors L est lui-même un ω-code générateur de L^ω. Sinon u et v ont la même racine primitive $\sqrt{u}$. Donc $\{\sqrt{u}\}$ est un ω-code générateur de L^ω. □

Exemple 10 (reprise de p. 15). Soit $L = \{a^2, a^3, b\}$. C'est un langage à délai 1 tel que L^+ est le plus grand générateur de L^ω. D'après la proposition 2.2.5, il n'existe pas de code infini qui soit un générateur de L^ω. De plus, l'ω-code fini $\{a^2, a^3b, b\}$ est un générateur de L^ω.

Exemple 14. Soit $L = \{a, ab, ba\}$. Le langage $a \cup (ab)^*ba$ est un ω-code infini générateur de L^ω. Maintenant on montre que L^ω n'a pas de code fini générateur.

Supposons que C soit un code fini générateur de L^ω et que n est le plus grand entier de l'ensemble $\{|u| : u \in C\}$. Comme $a^\omega \in C^\omega$, il existe un unique $i \geq 0$ tel que $a^{i+1} \in C$. On considère l'ω-mot

$$a^i(ab)^n(ba)^\omega \in C^\omega.$$

Donc ce mot a une factorisation cw où $c \in C$ et $w \in L^\omega$. Comme $b(ab)^*(ba)^\omega \cap C^\omega = \emptyset$, il existe $0 < j < n$ tel que $c = a^i(ab)^j \in C$. D'après la proposition 1.4.6 c'est impossible car l'ω-mot $(a^i(ab)^j)^\omega$ a deux C-factorisations :

$$(a^i(ab)^j)^\omega = a^i a((ba)^{j-1}ba^i a)^\omega.$$

On en déduit que L^ω n'a pas de code fini générateur.

Chapitre 3

Relateurs d'un langage et codes générateurs

Dans ce chapitre, nous étudions l'ensemble des relateurs satisfaits par un langage L, c'est-à-dire les couples (u, v) où u et v sont deux factorisations d'un même mot dans L^∞. Avec cette notation, nous établirons des conditions nécessaires pour qu'une ω-puissance ait un code ou un ω-code générateur.

3.1 Langages et relateurs

Soit $L \subseteq A^+$ un langage et soit Σ un alphabet qui a la même cardinalité que L, c'est-à-dire fini ou infini dénombrable. L'étiquetage des mots de L est une bijection $\tilde{}: \Sigma \to L$. Cette application se prolonge en un morphisme de $(\Sigma^\infty, \cdot)$ sur $(L^\infty, \cdot)$, où $\cdot$ dénote l'opération de concaténation.

On représentera donc une L-factorisation d'un mot de L^∞ par un mot de Σ^∞. Par abus de langage, les sous-ensembles de Σ^∞ sont appelés *langages des factorisations* sur L.

La relation $\cong$ sur Σ^∞ est définie en posant $x \cong y$ si et seulement si $\tilde{x} = \tilde{y}$. Un couple $x \cong y$ est dit *relateur* du langage L.

Définition 3.1.1. Un relateur $x \cong y$ est appelé *minimal* si $x \neq y$ et, pour tous

$u, v \in \Sigma^+$, on a

$$u < x \text{ et } v < y \quad \text{implique} \quad u \ncong v.$$

De façon équivalente, le relateur $x \cong y$ est minimal si et seulement si

$$\{\tilde{u} \in A^+ : u < x\} \cap \{\tilde{v} \in A^+ : v < y\} = \emptyset.$$

On note :

$$M(L) = \{(x, y) \in (\Sigma^\infty)^2 : x \cong y \text{ est un relateur minimal de } L\},$$
$$R(L) = \{(x, y) \in (\Sigma^\infty)^2 : x \cong y \text{ et } x \neq y\}.$$

Par définition, pour tout $u, v \in \Sigma^*$ et $u', v' \in \Sigma^\infty$, on a donc

$$\begin{cases} uu' \cong vv' \\ u \cong v \end{cases} \quad \text{implique} \quad u' \cong v'.$$

Ainsi $x \cong y$ si et seulement s'il existe une suite unique finie ou infinie de relateurs

$$(x_1 \cong y_1), \ldots, (x_n \cong y_n), \ldots$$

qui sont minimaux ou qui sont de la forme $z \cong z$ ($z \in \Sigma$) tels que $x = x_1 \ldots x_n \ldots$ et $y = y_1 \ldots y_n \ldots$.

Exemple 14 (reprise de p. 17). Soit $L = \{a, ab, ba\}$ et soit $\Sigma = \{0, 1, 2\}$ l'alphabet des relateurs de L. Les relateurs $02 \cong 10$ et $02^\omega \cong 1^\omega$ sont minimaux, mais $0210 \cong 1002$ n'est pas minimal. En effet, on peut vérifier que l'ensemble de relateurs minimaux de L est exactement le système suivant :

$$\begin{cases} 02^n \cong 1^n 0 & \text{pour tous } n > 0 \\ 02^\omega \cong 1^\omega \end{cases}$$

Remarque 3.1.2. Pour tout langage $C \subseteq \Sigma^+$, on a :

i) $\widetilde{C}$ est un code si et seulement si la condition

$$w_1 w_2 \dots w_n \;\cong\; v_1 v_2 \dots v_m$$

où $w_i, v_i \in C$ implique $m = n$ et $w_i \cong v_i$ pour tout $1 \leq i \leq n$.

ii) $\widetilde{C}$ est un ω-code si et seulement si la condition

$$w_1 w_2 \dots \;\cong\; v_1 v_2 \dots$$

où $w_i, v_i \in C$ implique $w_i \cong v_i$ pour tout $i \geq 1$.

Le lemme suivant nous donne une condition suffisante pour que l'image d'un langage sur Σ soit un code ou un ω-code.

Lemme 3.1.3. *i) Si $C \subseteq \Sigma^+$ est un code tel que $R(L) \cap (C^+)^2 = \emptyset$, alors $\widetilde{C}$ est un code.*

ii) Si $C \subseteq \Sigma^+$ est un ω-code tel que $R(L) \cap (C^\omega)^2 = \emptyset$, alors $\widetilde{C}$ est un ω-code.

Démonstration. On montre seulement $i)$; la démonstration de $ii)$ est similaire.

Supposons que $R(L) \cap (C^+)^2 = \emptyset$. Alors s'il existe un relateur

$$w_1 w_2 \dots w_n \;\cong\; v_1 v_2 \dots v_m$$

avec $w_i, v_i \in C$; alors ce relateur n'est pas dans $R(L)$. Donc on doit avoir

$$w_1 w_2 \dots w_n = v_1 v_2 \dots v_m.$$

Comme C est un code, on a $m = n$ et $w_i = v_i$ pour tout $n \geq i \geq 1$. □

On note $Amb_\Sigma(L)$ le sous-ensemble des ω-mots dans Σ^ω tels que les images de ces mots dans A^ω ont au moins deux L-factorisations. Ainsi,

$$\begin{aligned} Amb_\Sigma(L) &= \{x \in \Sigma^\omega : \tilde{x} \text{ a au moins deux } L\text{-factorisations}\} \\ &= \{x \in \Sigma^\omega : \exists y \in \Sigma^\omega,\ x \cong y \text{ et } x \neq y\}. \end{aligned}$$

Par définition, on a

$$\begin{aligned} Amb_\Sigma(L) &= \pi_1(R(L))\Sigma^\omega \\ &= \Sigma^* \, \pi_1(M(L))\Sigma^\omega \end{aligned}$$

où $\pi_1(R)$ dénote la première projection de la relation R.

D'après la proposition 1.4.6, L est un code si et seulement si $Amb_\Sigma(L)$ n'a pas de mots périodiques. Par définition, L est un ω-code si et seulement si $Amb_\Sigma(L) = \emptyset$.

3.2 Générateurs et codes

Dans cette section ainsi que la suivante, nous supposons que le langage L est tel que L^+ est le plus grand générateur de L^ω et que Σ est l'alphabet de relateurs de L.

Notons que cette hypothèse est satisfaite par des cas intéressants, par exemple, le cas où L^ω est finiment engendré ou, plus généralement, dans le cas où L^ω est une adhérence (d'après la proposition 2.1.6).

Soit $X \subseteq \Sigma^\infty$. On note $P(X) = \mathrm{Pref}(X) - \varepsilon$.

Lemme 3.2.1. *Soit $C \subseteq \Sigma^+$ tel que $\widetilde{C}$ est un générateur de L^ω et soit $w \in \Sigma^\omega$. Si $w \notin Amb_\Sigma(L)$, alors $w \in C^\omega$.*

Démonstration. Puisque $w \notin Amb_\Sigma(L)$, $\tilde{w}$ a une seule factorisation sur L. De plus, comme L^+ est le plus grand générateur de L^ω. Donc on a $w \in C^\omega$. □

Lemme 3.2.2. *Soit $C \subseteq \Sigma^+$ tel que $\widetilde{C}$ est un code générateur de L^ω. Alors le langage C est un code préfixe sur Σ.*

Démonstration. S'il existe deux mots $u, v \in \Sigma^+$ tels que $u, uv \in C$, alors on a $(uv)^\omega = (u)(vu)^\omega$. Comme $\widetilde{C}$ est un générateur de L^ω, il existe $w \in C^\omega$ tel que $w \cong (vu)^\omega$. Donc $(uv)^\omega \cong (u)w$ et $uv \ncong u$. D'après la proposition 1.4.6, $\widetilde{C}$ n'est pas un code. □

On dit que deux mots $u, v \in \Sigma^+$ sont *incompatibles* s'il existe $x, y \in \Sigma^+$ tels que le relateur $ux \cong vy$ est minimal.

Remarque 3.2.3. Si deux mots u et v sont incompatibles, alors u et v n'ont pas de préfixe commun.

Lemme 3.2.4. *Soit $C \subseteq \Sigma^+$ tel que $\widetilde{C}$ est un code générateur de L^ω et soient u et v deux mots incompatibles. Alors pour tout $m \in \Sigma^*$, l'ensemble $mP(\{u, v\}) \cap C$ est l'ensemble vide ou un singleton.*

Démonstration. D'après le lemme 3.2.2, C est un code préfixe. En conséquence, chacun des ensembles $mP(u) \cap C$ et $mP(v) \cap C$ est soit l'ensemble vide, soit un singleton. Maintenant supposons qu'il existe $m \in \Sigma^*$, $p \in P(u)$ et $q \in P(v)$ tels que $\{mp, mq\} \subseteq C$.

Comme u, v sont incompatibles, il existe $x, y \in \Sigma^+$ tels que

$$px \cong qy$$

est un relateur minimal.

Comme $\widetilde{C}$ est générateur de L^ω et d'après le lemme 1.4.4 l'ω-mot $(\tilde{x})^\omega$ a une L-factorisation ultimement périodique sur $\widetilde{C}$. Ainsi, il existe $z \in C^*$ et $t \in C^+$ tels que $x^\omega \cong zt^\omega$. D'après le lemme 1.4.5, il existe $i, j > 0$ tel que $x^i z \cong zt^j$. Maintenant on a

$$\begin{aligned}(mpzt^j)^\omega &\cong (mpxx^{i-1}z)^\omega \\ &\cong (mqyx^{i-1}z)^\omega = mq(yx^{i-1}zmq)^\omega.\end{aligned}$$

Comme $\widetilde{C}$ est un générateur de L^ω, il existe $w \in C^\omega$ tel que

$$(yx^{i-1}zmq)^\omega \cong w.$$

Donc on a

$$(mpzt^j)^\omega \cong (mq)w$$

On rappelle que C est un code préfixe et que $\{mp, mq\} \subseteq C$. D'après la proposition 1.4.6, on a donc $mp \cong mq$, alors $p \cong q$. C'est impossible car $px \cong qy$ est minimal. □

Voici un corollaire du lemme 3.2.4.

Corollaire 3.2.5. *Soit $C \subseteq \Sigma^+$ tel que $\widetilde{C}$ est un code générateur de L^ω et soient u et v deux mots incompatibles. Alors pour tout $m \in \Sigma^*$, il existe $z \in \{u, v\}$ tel que*

$$mP(z) \cap C = \emptyset.$$

Démonstration. D'après le lemme 3.2.4 et le remarque 3.2.3,on doit avoir

$$mP(u) \cap C = \emptyset \quad \text{ou} \quad mP(v) \cap C = \emptyset.$$

□

Proposition 3.2.6. *Soit $(\{u_i, v_i\})_{i\geq 0}$ une suite infinie de paires de mots incompatibles. Si*

$$Amb_\Sigma(L) \ \cap \ \prod_{i=0}^{\infty} \{u_i, v_i\} = \emptyset$$

où $\prod_{i=0}^{\infty} X_i$ représente la concaténation des langages X_i, alors L^ω n'a pas de code générateur.

Démonstration. Supposons qu'il existe $C \subseteq \Sigma^+$ tel que $\widetilde{C}$ soit un code générateur de L^ω. On va construire par induction une suite infinie de mots $(z_i)_{i\geq 0}$ telle que :

$$\begin{cases} z_i \in \{u_i, v_i\} \\ P(z_0 \dots z_i) \cap C = \emptyset, \end{cases} \tag{3.1}$$

pour tout $i \geq 0$.

D'abord, d'après le corollaire 3.2.5, il existe $z_0 \in \{u_0, v_0\}$ tel que

$$P(z_0) \cap C = \emptyset.$$

Supposons qu'on ait la suite $(z_0, \dots, z_{n-1})$ qui vérifie la condition (3.1). D'après le corollaire 3.2.5, il existe $z_n \in \{u_n, v_n\}$ tel que

$$z_0 \dots z_{n-1} P(z_n) \cap C = \emptyset$$

et par hypothèse d'induction on a

$$P(z_0 \dots z_{n-1}) \cap C = \emptyset.$$

Donc $z_0, \dots, z_n$ vérifient la condition (3.1).

Maintenant on considère l'ω-mot :

$$w = \prod_{i=0}^{\infty} z_i \notin Amb_\Sigma(L)$$

d'après le lemme 3.2.1, on a $w \in C^\omega$. Cependant, par construction on a $P(w) \cap C = \emptyset$. C'est une contradiction. □

La proposition suivante est un corollaire de la proposition 3.2.6.

Proposition 3.2.7. *Soient u et v deux mots incompatibles. Si*

$$Amb_\Sigma(L) \cap \{u, v\}^\omega = \emptyset,$$

alors L^ω n'a pas de code générateur.

Démonstration. Par application de la proposition 3.2.6 à la suite infinie de paires de mots incompatibles $(\{u, v\}, \{u, v\}, \cdots)$. □

Exemple 15. Soit $L = \{a, ab, bc, c\}$ et soit $\Sigma = \{0, 1, 2, 3\}$ l'alphabet des relateurs de L. L a un seul relateur minimal $02 \cong 13$ et alors $Amb_\Sigma(L) = \Sigma^*\{02, 13\}\Sigma^\omega$. On a

$$Amb_\Sigma(L) \cap \{0, 1\}^\omega = \emptyset.$$

D'après la proposition 3.2.7, L^ω n'a pas de code générateur.

Exemple 16. Soit $L = \{a, ab, ba, c, cd, dc\}$ et soit $\Sigma = \{0, 1, 2, 3, 4, 5\}$ l'alphabet des

relateurs de L. L'ensemble des relateurs minimaux de L est le système suivant :

$$\begin{cases} 02^n \cong 1^n 0 & \text{pour tous } n > 0 \\ 02^\omega \cong 1^\omega & \\ 35^m \cong 4^m 3 & \text{pour tous } m > 0 \\ 35^\omega \cong 4^\omega & \end{cases}$$

On considère la suite infinie de paires de mots incompatibles

$$(\{0,1\},\{3,4\},\{0,1\},\{3,4\},\cdots).$$

Comme

$$Amb_\Sigma(L) \cap (\{0,1\}\{3,4\})^\omega = \emptyset,$$

d'après la proposition 3.2.6, L^ω n'a pas de code générateur.

3.3 Générateurs et ω-codes

On dit que deux mots $u, v \in \Sigma^+$ sont ∞-*incompatibles* s'il existe $x, y \in \Sigma^+ \cup \Sigma^\omega$ tels que $ux \cong vy$ est minimal.

Remarque 3.3.1. Si deux mots u et v sont incompatibles, alors ils sont ∞-incompatibles.

Lemme 3.3.2. *Soit $C \subseteq \Sigma^+$ tel que $\widetilde{C}$ est un ω-code générateur de L^ω et soient u et v deux mots ∞-incompatibles. Alors pour tout $m \in \Sigma^*$, l'ensemble $mP(\{u,v\}) \cap C$ est vide ou singleton.*

Démonstration. D'après le lemme 3.2.2, C est un code préfixe. En conséquence, chacun des ensembles $mP(u) \cap C$ et $mP(v) \cap C$ est soit l'ensemble vide, soit un singleton. Maintenant supposons qu'il existe $m \in \Sigma^*$, $p \in P(u)$ et $q \in P(v)$ tels que $\{mp, mq\} \subseteq C$.

Comme u et v sont ∞-incompatibles, il existe $x, y \in \Sigma^\infty$ tels que

$$px \cong qy$$

est minimal. Donc on a $mpx \cong mqy$; et comme $\widetilde{C}$ est un ω-code générateur, on doit avoir $mp \cong mq$, alors $p \cong q$. C'est impossible car $px \cong qy$ est minimal. □

Les résultats suivants sont similaires à ceux sur les mots incompatibles.

Corollaire 3.3.3. *Soit $C \subseteq \Sigma^+$ tel que $\widetilde{C}$ est un ω-code générateur de L^ω et soient u, v deux mots ∞-incompatibles. Alors pour tout $m \in \Sigma^*$, il existe $z \in \{u, v\}$ tel que*

$$mP(z) \cap C = \emptyset.$$

Proposition 3.3.4. *Soit $(\{u_i, v_i\})_{i\geq 0}$ une suite infinie de paires de mots ∞-incompatibles. Si*

$$Amb_\Sigma(L) \cap \prod_{i=0}^{\infty} \{u_i, v_i\} = \emptyset$$

alors L^ω n'a pas d'ω-code générateur.

Proposition 3.3.5. *Soient u et v deux mots ∞-incompatibles. Si*

$$Amb_\Sigma(L) \cap \{u, v\}^\omega = \emptyset,$$

alors L^ω n'a pas d'ω-code générateur.

Exemple 3 (reprise de p. 6). Soit $L = \{a, ab, b^2\}$ et soit $\Sigma = \{0, 1, 2\}$ l'alphabet des relateurs de L. Le langage L est un code suffixe mais n'est pas un ω-code. En effet, L a un seul relateur minimal $02^\omega \cong 12^\omega$. Comme $Amb_\Sigma(L) \cap \{0, 1\}^\omega = \emptyset$, d'après la proposition 3.3.5, L^ω n'a pas d' ω-code générateur.

3.4 Calcul des relateurs minimaux d'un langage fini

Pour étudier les relateurs minimaux d'un langage fini L, on va considérer le graphe de domino de L qui est défini dans [Guz99] et [HW95]. Ce graphe est similaire au graphe de Spehner dans [Spe75, Lal79], le graphe des retards dans [Aug01] et le graphe d'ambiguïté dans [JT09].

On note $P = \mathrm{Pref}(L) - \varepsilon$. Soit $G = (V, E)$ un graphe orienté qui a l'ensemble des sommets suivants :

$$V = \{\texttt{open}, \texttt{close}\} \cup \left\{ \begin{bmatrix} u \\ \varepsilon \end{bmatrix} : u \in P \right\} \cup \left\{ \begin{bmatrix} \varepsilon \\ u \end{bmatrix} : u \in P \right\}$$

et l'ensemble des arêtes $E = E_1 \cup E_2 \cup E_3 \cup E_4$ où

$$\begin{aligned}
E_1 &= \left\{ \left(\texttt{open}, \begin{bmatrix} \varepsilon \\ u \end{bmatrix} \right) : u \in L \right\}, \\
E_2 &= \left\{ \left(\begin{bmatrix} u \\ \varepsilon \end{bmatrix}, \texttt{close} \right) : u \in L \right\}, \\
E_3 &= \left\{ \left(\begin{bmatrix} u \\ \varepsilon \end{bmatrix}, \begin{bmatrix} uv \\ \varepsilon \end{bmatrix} \right) : v \in L \right\} \cup \left\{ \left(\begin{bmatrix} \varepsilon \\ u \end{bmatrix}, \begin{bmatrix} \varepsilon \\ uv \end{bmatrix} \right) : v \in L \right\}, \\
E_4 &= \left\{ \left(\begin{bmatrix} u \\ \varepsilon \end{bmatrix}, \begin{bmatrix} \varepsilon \\ v \end{bmatrix} \right) : uv \in L \right\} \cup \left\{ \left(\begin{bmatrix} \varepsilon \\ u \end{bmatrix}, \begin{bmatrix} v \\ \varepsilon \end{bmatrix} \right) : uv \in L \right\}.
\end{aligned}$$

Un chemin fini de G est *réussi* lorsque son sommet de départ est `open` et son sommet d'arrivée est `close`. Un chemin infini de G est *réussi* lorsque son sommet de départ est `open`.

Le *graphe de domino* associé à L, noté $G(L)$, est le sous-graphe (V', E') de G où V' se compose des sommets $\texttt{open}, \texttt{close}$ et des sommets $v \in V$ tel qu'il existe un chemin réussi qui passe par v ; et E' se compose des arêtes $e \in E$ tel qu'il existe un chemin réussi qui passe par e.

Pour $u \in L$ et $x \in \Sigma$, on note $\hat{u} = x$ si $\tilde{x} = u$.

La *fonction de domino* associée à L est l'application

$$d : E \to (\Sigma \times \{\varepsilon\}) \cup (\{\varepsilon\} \times \Sigma)$$

définie sur :

– E_1 par $d\left(\texttt{open}, \begin{bmatrix} \varepsilon \\ u \end{bmatrix} \right) = \begin{bmatrix} \widehat{u} \\ \varepsilon \end{bmatrix}$,

– E_2 par $d\left(\begin{bmatrix} u \\ \varepsilon \end{bmatrix}, \texttt{close}\right) = \begin{bmatrix} \widehat{u} \\ \varepsilon \end{bmatrix}$,

– E_3 par $d\left(\begin{bmatrix} u \\ \varepsilon \end{bmatrix}, \begin{bmatrix} uv \\ \varepsilon \end{bmatrix}\right) = \begin{bmatrix} \varepsilon \\ \widehat{v} \end{bmatrix}$ et $d\left(\begin{bmatrix} \varepsilon \\ u \end{bmatrix}, \begin{bmatrix} \varepsilon \\ uv \end{bmatrix}\right) = \begin{bmatrix} \widehat{v} \\ \varepsilon \end{bmatrix}$,

– E_4 par $d\left(\begin{bmatrix} u \\ \varepsilon \end{bmatrix}, \begin{bmatrix} \varepsilon \\ v \end{bmatrix}\right) = \begin{bmatrix} \widehat{uv} \\ \varepsilon \end{bmatrix}$ et $d\left(\begin{bmatrix} \varepsilon \\ u \end{bmatrix}, \begin{bmatrix} v \\ \varepsilon \end{bmatrix}\right) = \begin{bmatrix} \varepsilon \\ \widehat{uv} \end{bmatrix}$.

La fonction de domino se prolonge par morphisme sur les chemins à $\Sigma^\infty \times \Sigma^\infty$ en posant, pour chaque chemin fini p constitué des arêtes $(e_1, e_2, \ldots, e_m)$,

$$d(p) = d(e_1)d(e_2)\ldots d(e_n) \in \Sigma^* \times \Sigma^*$$

et pour chaque chemin infini p constitué des arêtes $(e_1, e_2, \ldots)$,

$$d(p) = d(e_1)d(e_2)\cdots \in \Sigma^\omega \times \Sigma^\omega.$$

Le couple $d(p)$ est appelé *domino* associé au chemin p de G. Les première et seconde composantes de $d(p)$ sont respectivement notées $d_1(p)$ et $d_2(p)$.

Un chemin réussi p de $G(L)$ modélise deux factorisations d'un même mot dans L^∞. Les factorisations obtenues sont $d_1(p)$ et $d_2(p)$. Donc $d_1(p) \cong d_2(p)$ est un relateur de L. La proposition suivante [Guz99] montre que ces relateurs sont minimaux.

Proposition 3.4.1. *Soient $x, y \in \Sigma^\infty$, $x \cong y$ est un relateur minimal de L si et seulement s'il existe un chemin réussi p dans $G(L)$ tel que $d(p) = \left[\begin{smallmatrix} x \\ y \end{smallmatrix}\right]$ ou $d(p) = \left[\begin{smallmatrix} y \\ x \end{smallmatrix}\right]$.*

On note $\mathcal{D}(L) = \{d(p) : p \text{ est un chemin réussi de } G(L)\}$.

Exemple 14 (reprise de p. 17). Soit $L = \{a, ab, ba\}$ et soit $\Sigma = \{0, 1, 2\}$ l'alphabet des relateurs de L. Le graphe de domino $G(L)$ est montré dans la figure 3.1, on a alors :

$$\begin{aligned} \mathcal{D}(L) &= \begin{bmatrix} 0 \\ \varepsilon \end{bmatrix} \left(\begin{bmatrix} \varepsilon \\ 1 \end{bmatrix} \begin{bmatrix} 2 \\ \varepsilon \end{bmatrix}\right)^\infty \begin{bmatrix} \varepsilon \\ 1 \end{bmatrix} \begin{bmatrix} \varepsilon \\ 0 \end{bmatrix} \begin{bmatrix} 2 \\ \varepsilon \end{bmatrix} \\ &= \begin{bmatrix} 0 \\ \varepsilon \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix}^\infty \begin{bmatrix} 2 \\ 10 \end{bmatrix}. \end{aligned}$$

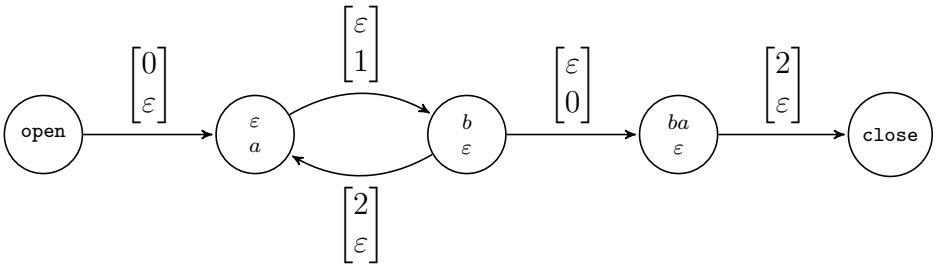


FIGURE 3.1 – Graphe de domino du langage $L = \{a, ab, ba\}$.

Donc l'ensemble de relateurs minimaux de L est exactement le système :

$$\begin{cases} 02^n \cong 1^n 0 & \text{pour tous } n > 0 \\ 02^\omega \cong 1^\omega \end{cases}$$

Le reste de cette section est consacré à montrer que chaque relateur minimal de L correspond à un seul chemin réussi de $G(L)$. D'abord on a besoin de lemmes techniques.

Lemme 3.4.2. *[Guz99] Soit L un langage de A^+. Soit $u \in L$ et p un chemin fini de* **open** *à $\left[\begin{smallmatrix} u \\ \varepsilon \end{smallmatrix}\right]$ (resp. $\left[\begin{smallmatrix} \varepsilon \\ u \end{smallmatrix}\right]$) de $G(L)$. Alors $\widetilde{d_1(p)}u = \widetilde{d_2(p)}$ (resp. $\widetilde{d_1(p)} = \widetilde{d_2(p)}u$).*

Lemme 3.4.3. *Soit L un langage de A^+. Soit $e \in E$ et $p = p'e$ un chemin fini de* **open** *à un sommet $\left[\begin{smallmatrix} u \\ v \end{smallmatrix}\right]$ de $G(L)$. Alors $\widetilde{d_2(p)} < \widetilde{d_1(p)}$ si et seulement si $d_2(e) = \varepsilon$.*

Démonstration. D'après le lemme 3.4.2, on a $\widetilde{d_2(p)} < \widetilde{d_1(p)}$ si et seulement si $u = \varepsilon$. D'après la définition de la fonction d, on a $d_2(e) = \varepsilon$ si et seulement si $u = \varepsilon$. □

Lemme 3.4.4. *Soient p et p' deux chemins (finis ou infinis) qui ont comme sommet de départ* **open**. *Si $d(p) = d(p')$, alors $p = p'$.*

Démonstration. On montre le lemme pour le cas où p et p' sont infinis. La preuve est similaire quand p et p' sont finis.

Posons

$$p = (\texttt{open} = w_0,\ w_1,\ w_2\ \ldots), \qquad p' = (\texttt{open} = z_0,\ z_1,\ z_2\ \ldots)$$

et

$$d(p) = d(p') = \begin{bmatrix} x_1 x_2 \dots \\ y_1 y_2 \dots \end{bmatrix}$$

où $w_i, z_i \in V$; $x_i, y_i \in \Sigma$. Supposons qu'il existe $k > 0$ tel que $w_0 = z_0, \dots, w_{k-1} = z_{k-1}$ et $w_k \neq z_k$. Posons

$$\begin{bmatrix} x_1 \dots x_i \\ y_1 \dots y_j \end{bmatrix} = d(w_0, \dots, w_{k-1}) = d(z_0, \dots, z_{k-1}) \tag{3.2}$$

D'après le lemme 3.4.2, on a :

$$d(w_{k-1}, w_k) \neq d(z_{k-1}, z_k) \tag{3.3}$$

Donc on peut supposer que :

$$d(w_{k-1}, w_k) = \begin{bmatrix} x_{i+1} \\ \varepsilon \end{bmatrix} \qquad \text{et} \qquad d(z_{k-1}, z_k) = \begin{bmatrix} \varepsilon \\ y_{j+1} \end{bmatrix} \tag{3.4}$$

On considère trois cas :

Cas 1 : $\tilde{x_1} \dots \tilde{x_{i+1}} = \tilde{y_1} \dots \tilde{y_{j+1}}$. C'est impossible à cause de (3.2) et (3.3).

Cas 2 : $\tilde{x_1} \dots \tilde{x_{i+1}} < \tilde{y_1} \dots \tilde{y_{j+1}}$. D'après (3.2) et (3.4), on a la décomposition

$$d(p') = \begin{bmatrix} x_1 \dots x_i \\ y_1 \dots y_j \end{bmatrix} \begin{bmatrix} \varepsilon \\ y_{j+1} \end{bmatrix} \begin{bmatrix} x_{i+1} \dots \\ y_{j+2} \dots \end{bmatrix}$$

Donc il existe un entier $\ell > 0$ tel que :

$$d(z_0, \dots, z_{k+l}) = \begin{bmatrix} x_1 \dots x_i \\ y_1 \dots y_{j+l} \end{bmatrix} \qquad \text{et} \qquad d(z_{k+l}, z_{k+l+1}) = \begin{bmatrix} x_{i+1} \\ \varepsilon \end{bmatrix}.$$

Comme $\tilde{x_1} \dots \tilde{x_{i+1}} < \tilde{y_1} \dots \tilde{y_{j+l}}$, d'après le lemme 3.4.3, on a $x_{i+1} = \varepsilon$. C'est impossible.

Cas 3 : $\tilde{y_1} \dots \tilde{y_{j+1}} < \tilde{x_1} \dots \tilde{x_{i+1}}$. Ce cas est symétrique du Cas 2. □

D'après la proposition 3.4.1 et le lemme 3.4.4, on a :

Proposition 3.4.5. *La fonction de domino d restreinte à l'ensemble des chemins réussis de $G(L)$ est une bijection sur l'ensemble des relateurs minimaux de L.*

3.5 Exemples

Exemple 17. Soit $L = \{a, ab, baba\}$ et soit $\Sigma = \{0, 1, 2\}$ l'alphabet des relateurs de L. Le langage

$$C = \{a, aba\} \cup (ab)^* baba$$

est un code générateur de L^ω. L'ensemble des relateurs minimaux de L est montré sur le graphe de domino de la figure 3.2. Donc on a

$$\mathcal{D}(L) = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} 2 \\ 11 \end{bmatrix}^\infty \left(\begin{bmatrix} \varepsilon \\ 0 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \end{bmatrix}^\omega \bigcup \begin{bmatrix} 2 \\ 10 \end{bmatrix} \right).$$

On en déduit que

$$Amb_\Sigma(L) = \Sigma^* \{02, 110\} \Sigma^\omega \cup \Sigma^* 1^\omega.$$

Comme $02^\omega \cong 102^\omega$ est un relateur minimal, les deux mots 0 et 10 sont ∞-incompatibles. De plus, on a

$$Amb_\Sigma(L) \cap \{0, 10\}^\omega = \emptyset$$

car $\mathrm{Fact}(\{0, 10\}^\omega) \cap \{11\} = \emptyset$. D'après la proposition 3.3.5, L^ω n'a pas d'ω-code générateur.

Exemple 18. Soit $L = \{a^2, a^3, b^2, b^3\}$ et soit $\Sigma = \{0, 1, 2, 3\}$ l'alphabet des relateurs de L. Comme $01 \cong 10$ et $23 \cong 32$ sont minimaux, les paires $\{0, 1\}$ et $\{2, 3\}$ sont des paires de mots incompatibles. De plus, il est clair que

$$Amb_\Sigma(L) \cap (\{0, 1\}\{2, 3\})^\omega = \emptyset,$$

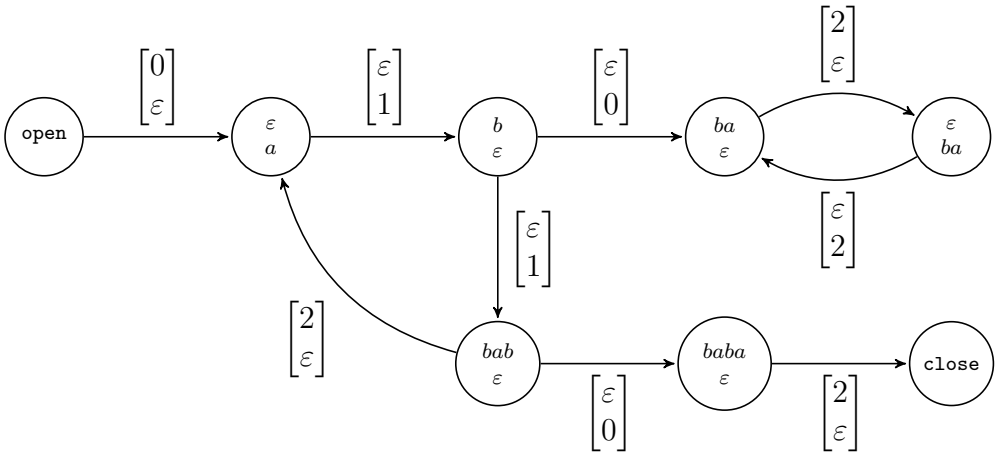


FIGURE 3.2 – Graphe de domino du langage $L = \{a, ab, baba\}$.

d'après la proposition 3.2.6, L^ω n'a pas de code générateur.

Ex.	Langage	code générateur	ω-code fini gén.	ω-code infini gén.
10	$\{a^2, a^3, b\}$	$\{a^2, a^3b, b\}$		non
14	$\{a, ab, ba\}$	$a \cup (ab)^*ba$	non	$a \cup (ab)^*ba$
3	$\{a, ab, b^2\}$	$\{a, ab, b^2\}$	non	
17	$\{a, ab, baba\}$	$\{a, aba\} \cup (ab)^*baba$	non	
18	$\{a^2, a^3, b^2, b^3\}$	non		
16	$\{a, ab, ba, c, cd, dc\}$	non		
15	$\{a, ab, bc, c\}$	non		

TABLE 3.1 – Récapitulatif des exemples.

Chapitre 4

Langages à un relateur et codes générateurs

On va appliquer les résultats du chapitre précédent pour traiter les problèmes 1 et 2 à savoir s'il existe un code ou un ω-code générateur d'une ω-puissance donnée. On sait (d'après le théorème 2.2.2) que si le plus grand générateur M de l'ω-puissance rationnelle L^ω est un semi-groupe libre, c'est-à-dire $M = L^+$ pour un code L, alors L^ω a un ω-code générateur si et seulement si L est lui-même un ω-code. Donc le problème 1 pour la classe des codes, c'est-à-dire pour la classe des langages à zéro relateur, est tout à fait résolu. On considère ici la classe des langages à un relateur.

En considérant que les codes sont les langages L où $L = L - L.L^+$ tels que L^+ est un semi-groupe libre, autrement dit ce sont les langages pour lesquels l'égalité de deux concaténations de mots de L, implique l'identité de ces deux concaténations, donc ce sont les langages pour lesquels il n'y a aucune égalité (autre que l'identité) entre deux concaténations, on essaye dans ce chapitre de définir les langages L où $L = L - L.L^+$ pour lesquels il y a « une seule » égalité entre deux concaténations différentes de mots de L. Or dès qu'il y a une telle égalité entre deux concaténations différentes, par exemple $u_1u_2 = v_1v_2v_3$, il y a une infinité d'autres à commencer par $(u_1u_2)^n = (v_1v_2v_3)^n$ pour tout entier $n > 0$ et $(u_1u_2)^\omega = (v_1v_2v_3)^\omega$. Par ailleurs comme on le verra, si il y a des chevauchements entre les deux concaténations, par exemple il y a un chevauchement si $u_1 = v_3$, il y a d'autres égalités non triviales provenant de ces chevauchements,

$u_1{u_2}^n = (v_1v_2)^n v_3$ pour tout entier $n > 0$ et $u_1{u_2}^\omega = (v_1v_2)^\omega$. Donc il n'y a jamais « une seule » égalité entre deux concaténations différentes, mais il existe des langages L pour lesquels toutes les égalités entre deux concaténations différentes de mots de L sont conséquences (par remplacements, concaténations ou passages à la limite) d'une unique égalité de base. C'est cette famille de langages à un relateur que l'on tente de définir ici et pour laquelle on arrive à résoudre les problèmes précédents.

4.1 Définitions et résultats

Soit L un langage (fini ou infini) sur l'alphabet A et soit $\Sigma = \{0, 1, 2, \ldots\}$ un autre alphabet qui est en bijection avec L comme cela a été fait dans le chapitre précédent.

Soit $\sim$ une relation d'équivalence sur le monoïde Σ^∞. On dit que $\sim$ est une *congruence* sur Σ^∞ si, pour tous $u_1, u_2, v_1, v_2 \in \Sigma^\infty$, on a

$$u_1 \sim v_1 \quad \text{et} \quad u_2 \sim v_2 \qquad \text{implique} \qquad u_1u_2 \sim v_1v_2.$$

Pour chaque relation R sur Σ^∞, il existe une plus petite congruence $R^\#$ sur Σ^∞ contenant R (voir [How95] page 25) ; c'est l'intersection des congruences contenant R :

$$R^\# = \bigcap \{\rho : R \subseteq \rho \ \text{ et } \ \rho \text{ est une congruence sur } \Sigma^\infty\}.$$

Pour un couple $(u, v) \in \Sigma^\infty \times \Sigma^\infty$, on note

$$\mathrm{Pref}(u, v) = \mathrm{Pref}(u) \times \mathrm{Pref}(v),$$

et pour une relation $R \subseteq \Sigma^\infty \times \Sigma^\infty$, on note $\mathrm{Pref}(R) = \bigcup_{(u,v)\in R} \mathrm{Pref}(u, v)$.

L'*adhérence* d'une relation $R \subseteq \Sigma^\infty \times \Sigma^\infty$ est définie dans [Niv81, Pri01] :

$$Adh(R) = \{(x, y) \in \Sigma^\omega \times \Sigma^\omega : \mathrm{Pref}(x, y) \subseteq \mathrm{Pref}(R)\}.$$

La relation R est *fermée* si $Adh(R) \subseteq R$.

Proposition 4.1.1. *L'intersection d'une famille de congruences fermées est une congruence fermée.*

Démonstration. Soit $(R_i)_{i\in I}$ une famille de congruences fermées. Alors $\cap_{i\in I} R_i$ est une congruence. D'autre part, pour tout $i \in I$, on a $Adh(\cap_{i\in I} R_i) \subseteq Adh(R_i)$. Donc $Adh(\cap_{i\in I} R_i) \subseteq \cap_{i\in I} Adh(R_i)$. Comme R_i est fermée pour tout $i \in I$, on a $\cap_{i\in I} Adh(R_i) \subseteq \cap_{i\in I} R_i$. Donc on a $Adh(\cap_{i\in I} R_i) \subseteq \cap_{i\in I} R_i$. □

D'après la proposition 4.1.1, pour chaque relation R, il existe une plus petite congruence fermée $\hat{R}$ contenant R ; $\hat{R}$ est l'intersection de toutes les congruences fermées contenant R.

Proposition 4.1.2. *L'ensemble des relateurs $\cong$ d'un langage L est une congruence fermée.*

Démonstration. Par définition, l'ensemble des relateurs $\cong$ est donc une congruence. De plus, soit

$$(x, y) = (x_0x_1 \dots, y_0y_1 \dots) \in Adh(\cong)$$

avec $x_i, y_i \in \Sigma$; alors $\mathrm{Pref}(x, y) \subseteq \mathrm{Pref}(\cong)$. Ainsi pour tout entier $i \geq 0$, il existe des mots $u_i, v_i \in \Sigma^\infty$ tels que

$$x_0 \dots x_i u_i \ \cong \ y_0 \dots y_i v_i.$$

Posons

$$p_i = \begin{cases} \tilde{x_0} \dots \tilde{x_i} & \text{si } |\tilde{x_0} \dots \tilde{x_i}| < |\tilde{y_0} \dots \tilde{y_i}| \\ \tilde{y_0} \dots \tilde{y_i} & \text{sinon.} \end{cases}$$

On a donc

$$p_0 < p_1 < \cdots < \tilde{x} \quad \text{et} \quad p_0 < p_1 < \cdots < \tilde{y}.$$

Alors $\tilde{x}$ et $\tilde{y}$ ont une infinité de préfixes communs. Donc on a $\tilde{x} = \tilde{y}$, c'est-à-dire $x \ \cong \ y$. En conséquence $\cong$ est fermée. □

Définition 4.1.3. On dit que $L = Rac(L^*)$ est un *langage à un relateur* s'il existe un couple $(u, v) \in \Sigma^+ \times \Sigma^+$, $u \neq v$ tel que la relation $\cong$ de L est la plus petite congruence fermée contenant (u, v). Alors le relateur $u \ \cong \ v$ est dit *relateur de base* de L.

Fait 4.1.4. *Pour un langage à un relateur, il existe un seul relateur de base (au symétrique près, c'est-à-dire si (u, v) est un relateur de base, alors (v, u) l'est aussi).*

Démonstration. Soit L un langage à un relateur et soient $u \cong v$ et $x \cong y$ deux relateurs de base de L. Par définition, on a

$$(u, v) \in \{(x, y)\}^{\#} \quad \text{et} \quad (x, y) \in \{(u, v)\}^{\#}.$$

Alors on a

$$u, v \in \mathrm{Fact}(\{x, y\}) \quad \text{et} \quad x, y \in \mathrm{Fact}(\{u, v\}).$$

Alors on a : soit $x = u$ et $y = v$, soit $x = v$ et $y = u$. □

Ainsi, on dit simplement que L est un langage à un relateur $u \cong v$ si L est un langage à un relateur et si $u \cong v$ est le relateur de base de L.

Comme tout relateur d'un langage peut s'obtenir par la concaténation de relateurs minimaux ou relateurs de la forme $z \cong z$ avec $z \in \Sigma$, on a :

Fait 4.1.5. *Un langage est à un relateur si et seulement s'il existe un relateur minimal $u \cong v$ tel que tout relateur minimal est dans la plus petite congruence fermée contenant (u, v).*

Exemple 15 (reprise de p. 25). Soient $L = \{a, ab, bc, c\}$ et $\Sigma = \{0, 1, 2, 3\}$. Le graphe de domino de L dans la figure 4.1 montre que L a un seul relateur minimal $02 \cong 13$. Donc L est un langage à un relateur $02 \cong 13$.

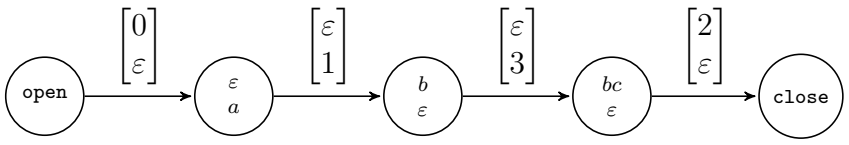


FIGURE 4.1 – Graphe de domino du langage $\{a, ab, bc, c\}$.

Exemple 14 (reprise de p. 17). Soient $L = \{a, ab, ba\}$ et $\Sigma = \{0, 1, 2\}$. L'ensemble des relateurs minimaux de L est exactement le système suivant :

$$\begin{cases} 02^n \cong 1^n 0 & \text{pour tout } n > 0 \\ 02^\omega \cong 1^\omega. \end{cases}$$

Montrons que L est un langage à un relateur $02 \cong 10$. En effet, comme $02 \cong 10$, par réécriture, on a $022 \cong 102 \cong 110$, on obtient $02^2 \cong 1^2 0$. Par récurrence, on obtient donc $02^n \cong 1^n 0$ pour tout $n > 0$. De plus, on a

$$Adh(\{02^n \cong 1^n 0 : n > 0\}) = \{02^\omega \cong 1^\omega\}.$$

Tout relateur de L est dans la plus petite congruence fermée contenant $(02, 10)$.

Exemple 19. Soient $L = \{ab, aba, b\}$ et $\Sigma = \{0, 1, 2\}$. Le graphe de domino de L est donné dans la figure 4.2. L'ensemble des relateurs minimaux de L est le système suivant :

$$\begin{cases} 0(12)^n 0 \cong (12)^n 12 & \text{pour tout } n \geq 0 \\ 0(12)^m \cong (12)^m 0 & \text{pour tout } m > 0 \\ 0(12)^\omega \cong (12)^\omega. \end{cases}$$

À partir du relateur $00 \cong 12$, on peut obtenir d'autres relateurs par réécriture :

$$\begin{aligned} 0(12)^n 0 &\cong 0(00)^n 0 = (00)^n 00 \cong (12)^n 12, \\ 0(12)^m &\cong 0(00)^m = (00)^m 0 \cong (12)^m 0. \end{aligned}$$

De plus, on a

$$Adh\left(\{0(12)^m \cong (12)^m 0 : m > 0\}\right) = \{0(12)^\omega \cong (12)^\omega\}.$$

Ainsi L est un langage à un relateur $00 \cong 12$.

Exemple 20. Soient $L = \{a, ab, ba, ac, ca\}$ et $\Sigma = \{0, 1, 2, 3, 4\}$. L n'est pas un langage

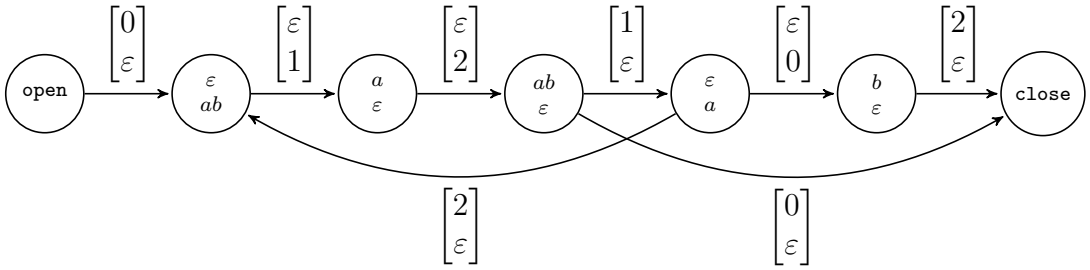


FIGURE 4.2 – Graphe de domino du langage $\{ab, aba, b\}$.

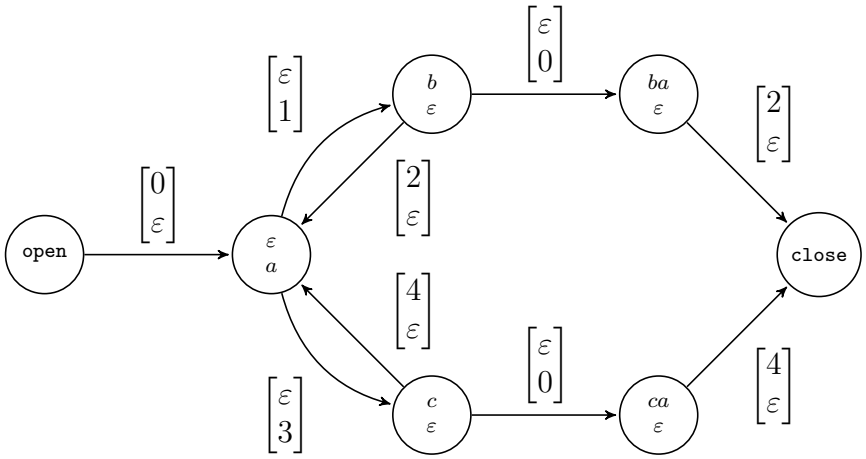


FIGURE 4.3 – Graphe de domino du langage $\{a, ab, ba, ac, ca\}$.

à un relateur car on a deux relateurs $02 \cong 10$ et $04 \cong 30$; et l'un ne peut pas se réécrire en l'autre. En effet, comme montré dans la figure 4.3, l'ensemble des relateurs de L est une congruence fermée contenant deux relateurs $02 \cong 10$ et $04 \cong 30$.

L'objectif de ce chapitre est de démontrer les deux résultats suivants.

Théorème 4.1.6. *Soit L un langage à un relateur tel que L^+ est le plus grand générateur de L^ω. Alors L^ω n'a pas de code fini générateur.*

Théorème 4.1.7. *Soit L un langage à un relateur tel que L^+ est le plus grand générateur de L^ω. Alors L^ω a un code générateur si et seulement si le relateur de base $u \cong v$ est d'une des formes suivantes :*

(i) $u = 0^n w 2$ et $v = 10^n$ avec $w \in \Sigma^*$ et $n \geq 1$;

(ii) $u = 0^n 2$ *et* $v = 10^m 0^n$ *avec* $m \geq 1$ *et* $n \geq 1$ *;*

(iii) $u = 0w2$ *et* $v = 1^k 0$ *avec* $w \in \Sigma^*$ *et* $k > 1$ *;*

(iv) $u = 02$ *et* $v = (10^m)^k 0$ *avec* $m \geq 1$ *et* $k > 1$.

En outre, L^ω *a un* ω*-code générateur si et seulement si le relateur de base est de la forme (i) ou (ii).*

Remarque 4.1.8. D'après le corollaire 4.2.7 et le lemme 4.2.8, dans l'item (iii) du théorème 4.1.7, on a $w \notin \{0,1\}^* 1 \Sigma^*$. Dans l'item (i), on a $w \in (\Sigma - 1)^*$ pour des raisons de longueur.

4.2 Lemmes préliminaires

On présente ici des lemmes techniques utilisés par la suite.

Soient u et v deux mots finis non vides. On note $\mathrm{OVL}(u, v)$ l'ensemble des chevauchements propres du mot u par le mot v, c'est-à-dire :

$$\mathrm{OVL}(u, v) = \{x \in \Sigma^+ : u = xz,\ v = yx \ \text{ pour } y, z \in \Sigma^+\}.$$

Un couple $(y, z) \in \Sigma^+ \times \Sigma^+$ est un *bord* du couple (u, v) s'il existe $x \in \mathrm{OVL}(u, v)$ tel que

$$u = xz \qquad \text{et} \qquad v = yx.$$

On dénote $\mathrm{BO}(u, v)$ l'ensemble des bords du (u, v).

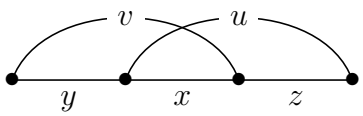


FIGURE 4.4 – $x \in \mathrm{OVL}(u, v)$ et $(y, z) \in \mathrm{BO}(u, v)$.

Notons que ces notations ne sont pas symétriques.

Exemple 21. Soient $u = 0102$ et $v = 1010$. On a

$$\begin{aligned} \mathrm{OVL}(u,v) &= \{010, 0\} & \mathrm{BO}(u,v) &= \{(1,2),(101,102)\} \\ \mathrm{OVL}(v,u) &= \emptyset & \mathrm{BO}(v,u) &= \emptyset. \end{aligned}$$

Lemme 4.2.1. *Soit L un langage à un relateur $u \cong v$. L'ensemble des relateurs de L est égal à la plus petite congruence contenant M où*

$$M = (\varepsilon, u) \cdot \left(\bigcup_{(u_1,u_2)\in\{u,v\}^2} \mathrm{BO}(u_1,u_2) \right)^{\infty} \cdot (v, \varepsilon).$$

Démonstration. Montrons d'abord que $M^{\#} \subseteq \cong$. On note

$$B = \bigcup_{(x,y)\in\{u,v\}^2} \mathrm{BO}(x,y)$$

Si $B = \emptyset$, alors le résultat est immédiat car $u \cong v$. Supposons donc $B \neq \emptyset$. On montre d'abord que pour tout $(y,z) \in B$, on a

$$uz \cong yv. \tag{4.1}$$

On vérifie (4.1) dans les quatre cas :

(1) Si $(y,z) \in \mathrm{BO}(u,u)$. Par définition, il existe $x \in \Sigma^+$ tels que $u = yx = xz$. On a donc $uz = yxz = yu \cong yv$.

(2) Si $(y,z) \in \mathrm{BO}(u,v)$. Par définition, il existe $x \in \Sigma^+$ tels que $v = yx$ et $u = xz$. On a donc $uz \cong vz = yxz = yu \cong yv$.

(3) Si $(y,z) \in \mathrm{BO}(v,u)$. Par définition, il existe $x \in \Sigma^+$ tels que $u = yx$ et $v = xz$. On a donc $uz = yxz = yv$.

(4) Si $(y,z) \in \mathrm{BO}(v,v)$. Par définition, il existe $x \in \Sigma^+$ tels que $v = yx = xz$. On a donc $uz \cong vz = yxz = yv$.

Ensuite, on considère une suite infinie

$$(y_1, z_1), (y_2, z_2), \ldots$$

avec pour tout i, $(y_i, z_i) \in B$. D'après (4.1), on a $uz_i \cong y_i v$. Ainsi pour tout entier positif n, on a

$$(uz_1)z_2 \ldots z_n \cong y_1(vz_2)z_3 \ldots z_n \cong y_1(uz_2)z_3 \ldots z_n \cong \ldots \cong y_1 \ldots y_{n-1}y_n v.$$

Comme la relation $\cong$ est fermée, par passage à l'adhérence, on obtient

$$uz_1z_2 \ldots \cong y_1y_2 \ldots.$$

Alors on a $M \subseteq \cong$.

On montre maintenant l'inclusion $\cong \subseteq M^{\#}$. Tout d'abord, on a

$$\{(pw, w) : (p^\omega, w) \in M \text{ ou } (w, p^\omega) \in M\} \subseteq M^{\#}$$

car si $(p^\omega, w) = (pp^\omega, w) \in M$ alors $(pw, w) \in M^{\#}$. Donc la congruence $M^{\#}$ est fermée et $(v, u) \in M^{\#}$, par la définition de langage à un relateur, on a le résultat. □

Exemple 17 (reprise de p. 32). Soit $L = \{a, ab, baba\}$ et soit $\Sigma = \{0, 1, 2\}$ l'alphabet des relateurs de L. L'ensemble des relateurs minimaux de L est le système suivant :

$$\begin{cases} 02^n \cong (11)^n 0 & \text{pour tout } n > 0 \\ 02^\omega \cong 1^\omega & \\ 02^\omega \cong 1(11)^m 02^\omega & \text{pour tout } m \geq 0 \end{cases}$$

On peut vérifier que les relateurs $02^n \cong (11)^n 0$ et $02^\omega \cong 1^\omega$ sont obtenus par réécriture ou par passage à l'adhérence à partir du relateur $02 \cong 110$. De plus, pour tout $m \geq 0$, on a

$$02^\omega \cong 1^\omega = 1(11)^m 1^\omega \cong 1(11)^m 02^\omega.$$

Alors L est un langage à un relateur $02 \cong 110$.

On note

$$\mathrm{LB}(u,v) = v\,(\mathrm{OVL}(u,v))^{-1} = \pi_1(\mathrm{BO}(u,v))$$

où π_1 dénote la première projection.

Voici un corollaire du lemme 4.2.1.

Lemme 4.2.2. *Soit L un langage à un relateur avec le relateur de base $u \cong v$. On a*

$$Amb_\Sigma(L) = \Sigma^*\{u,v\}\Sigma^\omega \cup \Sigma^*\Big(\mathrm{LB}(u,u) \cup \mathrm{LB}(v,v) \cup \mathrm{LB}(u,v) \cup \mathrm{LB}(v,u)\Big)^\omega.$$

Si $\mathrm{OVL}(u,v) \neq \emptyset$, alors $\mathrm{OVL}(u,v)$ a un plus grand élément (pour la longueur), noté $O_{u,v}$ et $\mathrm{LB}(u,v)$ a un plus petit élément (pour la longueur), noté $B_{u,v}$. On a donc

$$v = B_{u,v}O_{u,v}.$$

Exemple 22. Soient $u = 0102$ et $v = 1010$. On a

$$\mathrm{OVL}(u,v) = \{010, 0\} \qquad \mathrm{LB}(u,v) = \{1, 101\}$$
$$O_{u,v} = \{010\} \qquad B_{u,v} = \{1\}.$$

Pour un mot $u \in \Sigma^\infty$, on note $\mathrm{Alph}(u)$ l'ensemble des lettres de Σ qui apparaissent au moins une fois dans u.

Fait 4.2.3. *Soit $u \in \Sigma^+$. Pour tout $y \in \mathrm{LB}(u,u)$, on a $\mathrm{Alph}(y) = \mathrm{Alph}(u)$.*

Démonstration. Pour tout $y \in \mathrm{LB}(u,u)$, il existe deux mots non vides x et z tels que $u = xz = yx$. D'après le théorème 1.1.3, il existe deux mots α, β et un entier k tels que $x = (\alpha\beta)^k\alpha$, $y = \alpha\beta$ et $z = \beta\alpha$. Donc on a $\mathrm{Alph}(u) = \mathrm{Alph}(\alpha\beta) = \mathrm{Alph}(y)$. □

Fait 4.2.4. *Pour tout mot $u \in \Sigma^+$, on a $\mathrm{LB}(u,u)^\omega \subseteq u\Sigma^\omega$.*

Démonstration. Si $\mathrm{OVL}(u,u) = \emptyset$, le résultat est immédiat. Supposons maintenant que $\mathrm{OVL}(u,u) = \{x_1, \ldots, x_n\}$ avec $n > 0$. Pour tout $n \geq i \geq 1$, on peut donc écrire u sous la forme $u = y_ix_i = x_iz_i$ où $y_i \in \mathrm{LB}(u,u)$; on a alors $uz_i = y_ix_iz_i = y_iu$. Par

récurrence, pour toute suite (i_j) avec $n \geq i_j \geq 1$, on a

$$uz_{i_1}z_{i_2}\dots z_{i_k} = y_{i_1}y_{i_2}\dots y_{i_k}u.$$

Par passage à l'infini, on obtient

$$uz_{i_1}z_{i_2}\dots z_{i_k}\cdots = y_{i_1}y_{i_2}\dots y_{i_k}\dots.$$

Il s'ensuit que $\mathrm{LB}(u,u)^\omega \subseteq u\Sigma^\omega$. □

Fait 4.2.5. *Soient $u, v \in \Sigma^+$. Si $\mathrm{OVL}(u,v) \neq \emptyset$, alors pour tout $w \in \mathrm{LB}(u,v) - B_{u,v}$, on a $\mathrm{Alph}(w) = \mathrm{Alph}(v)$.*

Démonstration. Par définition, on a

$$\mathrm{OVL}(u,v) = \{O_{u,v}\} \cup \mathrm{OVL}(O_{u,v}, O_{u,v}).$$

Donc

$$\mathrm{LB}(u,v) = \{B_{u,v}\} \cup \{B_{u,v}\}\,\mathrm{LB}(O_{u,v}, O_{u,v}).$$

D'après le fait 4.2.3, pour tout $y \in \mathrm{LB}(O_{u,v}, O_{u,v})$, on a $\mathrm{Alph}(y) = \mathrm{Alph}(O_{u,v})$; on obtient donc

$$\mathrm{Alph}(B_{u,v}y) = \mathrm{Alph}(B_{u,v}O_{u,v}) = \mathrm{Alph}(u),$$

ce qui achève la preuve du fait. □

On note respectivement $\mathrm{First}(x)$ et $\mathrm{Last}(x)$ la première lettre et la dernière lettre du mot non vide x.

Lemme 4.2.6. *Si L est un langage à un relateur, alors le relateur de base de L ne peut pas s'écrire sous la forme $uv \cong vu$ avec $u, v \in \Sigma^*$.*

Démonstration. Supposons au contraire que L est un langage à un relateur avec le relateur de base $uv \cong vu$. Ainsi $\tilde{u}$ et $\tilde{v}$ sont la puissance d'un même mot, alors il existe deux entiers positifs p et q tels que $u^p \cong v^q$. Alors le couple (u^p, v^q) appartient à la plus petite congruence qui contient (uv, vu). Ainsi u^p contient un facteur uv ou

un facteur vu, autrement dit, il existe deux mots $x, y \in \Sigma^*$ tels que $u^p = xuvy$ ou $u^p = xvuy$. Les deux cas mènent à une contradiction :

- si $u^p = xuvy$, alors $uu^p = uxuvy = u^pu = xuvyu$. Comme $|xu| = |ux|$, on a $uvy = vyu$. C'est impossible car $\mathrm{First}(u) \neq \mathrm{First}(v)$.
- si $u^p = xvuy$, alors $uu^p = uxvuy = u^pu = xvuyu$. Donc on a $uxv = xvu$. C'est impossible car $\mathrm{Last}(u) \neq \mathrm{Last}(v)$. □

D'après le théorème 1.1.2, on a le corollaire suivant.

Corollaire 4.2.7. *Si L est un langage à un relateur, alors L ne contient pas deux mots qui commutent.*

Les deux lemmes suivants sont souvent utilisés pour montrer qu'un langage contient deux mots qui commutent et donc n'est pas à un relateur.

Lemme 4.2.8. *[CK07] Soient $x, y \in A^+$ et soient $z, t \in \{x, y\}^*$. Si xzy et ytx sont tous les deux préfixes ou tous les deux suffixes d'un même mot, alors x et y commutent.*

Lemme 4.2.9. *Soient $x, y, t \in A^*$ et soit $z \in \{x, y\}^*$. Si $xzy = ytx$, alors x et y commutent.*

Démonstration. Si $x = \varepsilon$ ou $y = \varepsilon$, alors x et y commutent. On raisonne par récurrence sur la longueur $|xy|$. On suppose que le résultat est vrai pour tous mots x, y où $|xy| < n$. On le prouve pour $|xy| = n$.

Si $|x| = |y|$ alors $x = y$. Comme x et y jouent un rôle symétrique par l'image miroir, on peut supposer que $|x| < |y|$. Alors x est un préfixe propre de y, on écrit $y = xx'$ avec $x' \in A^+$; on a $xzxx' = xx'tx$, alors $zxx' = x'tx$. Comme $zx \in \{x, xx'\}^*x \subseteq x\{x, x'\}^*$, on pose $z' = x^{-1}zx \in \{x, x'\}^*$ alors on a $xz'x' = x'tx$. De plus, si $x \neq \varepsilon$, alors $|xx'| < |xy|$; par hypothèse de récurrence, x et x' commutent. Donc $y = xx'$ et x commutent. □

Remarque 4.2.10. On ne peut pas supprimer la condition $z \in \{x, y\}^*$ dans le lemme 4.2.9. Par exemple si $x = a, y = aba, z = t = b$, on a $xzy = ytx$ mais x et y ne commutent pas.

Lemme 4.2.11. *Soient* 0 *et* 1 *deux lettres distinctes dans* Σ *et soit* x *un mot sur* Σ. *Il s'avère que* $x0 \notin \mathrm{Fact}(\{x1, 1\}^\omega)$.

Démonstration. Pour tout $y \in \mathrm{Fact}(\{x1, 1\}^\omega)$ tel que $|y| = |x| + 1$, il est clair que $|y|_1 > |x|_1$. Or $|x0|_1 = |x|_1$ donc $x0 \notin \mathrm{Fact}(\{x1, 1\}^\omega)$. □

Le lemme suivant [LT87] dit « lemme d'itération infinie » est souvent utilisé pour montrer l'égalité de deux ω-puissances.

Lemme 4.2.12. *Soient* L *et* R *deux langages sur* A. *Si* $L^\omega \subseteq RL^\omega$ *alors* $L^\omega \subseteq R^\omega$.

4.3 Démonstration du théorème 4.1.7

On rappelle que L est langage à un relateur $u \cong v$. Par minimalité de ce relateur $\mathrm{First}(u) \neq \mathrm{First}(v)$. Ainsi on peut supposer que $\mathrm{First}(u) = 0$ et $\mathrm{First}(v) = 1$ où 0 et 1 sont deux lettres distinctes de Σ.

Pour démontrer le théorème 4.1.7, on utilise six propriétés 4.3.1–4.3.8 que l'on va démontrer dans les sous-sections suivantes.

Comme u et v jouent un rôle symétrique, on considère trois cas :

Cas A : $\mathrm{OVL}(u, v) = \emptyset$ et $\mathrm{OVL}(v, u) = \emptyset$.

Propriété 4.3.1. *Si* $\mathrm{OVL}(u, v) = \emptyset$ *et* $\mathrm{OVL}(v, u) = \emptyset$, *alors* L^ω *n'a pas de code générateur.*

Cas B : $\mathrm{OVL}(u, v) \neq \emptyset$ et $\mathrm{OVL}(v, u) \neq \emptyset$.

Propriété 4.3.2. *Si* $\mathrm{OVL}(u, v) \neq \emptyset$ *et* $\mathrm{OVL}(v, u) \neq \emptyset$, *alors* L^ω *n'a pas de code générateur.*

Cas C : $\mathrm{OVL}(u, v) \neq \emptyset$ et $\mathrm{OVL}(v, u) = \emptyset$.

Notation 4.3.3. *On réécrit le relateur de base* $u \cong v$ *sous la forme*

$$0xz \cong 1y0x \tag{4.2}$$

où $z \in \Sigma^+$ *;* $x, y \in \Sigma^*$ *; et* $0x = O_{u,v}$ *est le plus grand élément (pour la longueur) de* $\mathrm{OVL}(u, v)$.

Fait 4.3.4. *Soit le relateur de base de L sous la forme* (4.2)*. On a*

$$xz \notin \{0,1\}^+ \tag{4.3}$$

Démonstration. Supposons au contraire que $x \in \{0,1\}^*$ et $z \in \{0,1\}^+$. Comme $0xz \cong 1y0x$ est le relateur de base, on a $\mathrm{Last}(z) \neq \mathrm{Last}(0x)$.

Si $\mathrm{Last}(0x) = 0$ alors on a $\mathrm{Last}(z) = 1$, contradiction car $\mathrm{OVL}(v, u) = \emptyset$.

Si $\mathrm{Last}(0x) = 1$ alors on a $\mathrm{Last}(z) = 0$. Donc on peut écrire le relateur de base sous la forme

$$0x'1z'0 \cong 1y0x'1$$

où $x'1 = x$ et $z'0 = z$. Ainsi $\widetilde{1z'0}$ et $\widetilde{0x'1}$ sont suffixes d'un même mot. D'après le lemme 4.2.8, alors $\tilde{0}$ et $\tilde{1}$ commutent. D'après le corollaire 4.2.7, contradiction avec le fait que L est un langage à un relateur. □

Propriété 4.3.5. *Soit le relateur de base de L sous la forme* (4.2)*. Si* $x \notin 0^*$ *ou* $\sqrt{1y} \notin 10^*$*, alors* L^ω *n'a pas de code générateur.*

Il reste à étudier les cas où $0x = 0^n$ et $1y = (10^m)^k$ avec $k \geq 1, m \geq 0$ et $n \geq 1$. Comme $\mathrm{OVL}(v, u) = \emptyset$, on a $\mathrm{Last}(z) \notin \{0,1\}$.

Propriété 4.3.6. *Soit le relateur de base de L sous la forme*

$$0^n z \cong (10^m)^k 0^n \quad \textit{avec} \quad 0^n = O_{u,v} \textit{ et } \mathrm{Last}(z) \notin \{0,1\}.$$

Si $k \geq 2, m \geq 0$ *et* $n \geq 2$*, alors* L^ω *n'a pas de code générateur.*

Il reste à étudier les cas où $k = 1$ ou $n = 1$. De plus, on a

Propriété 4.3.7. *Soit le relateur de base de L sous la forme*

$$0^n z \cong (10^m)^k 0^n \quad \textit{avec} \quad 0^n = O_{u,v} \textit{ et } \mathrm{Last}(z) \notin \{0,1\}.$$

Si $|z| \geq 2$, $k \geq 1, m \geq 1$ *et* $n \geq 1$, *alors* L^ω *n'a pas de code générateur.*

D'après les propriétés 4.3.5–4.3.7, il reste donc à montrer que L^ω a toujours un code générateur dans les cas où le relateur de base est sous la forme

$$0^n z \ \cong \ (10^m)^k 0^n$$

avec

(i) ou bien $k = 1, m = 0, n \geq 1$ et $z \in \Sigma^* 2$;

(ii) ou bien $k = 1, m \geq 1, n \geq 1$ et $z = 2$;

(iii) ou bien $k > 1, m = 0, n = 1$ et $z \in \Sigma^* 2$;

(iv) ou bien $k > 1, m \geq 1, n = 1$ et $z = 2$.

Cas (i) : $k = 1, m = 0, n \geq 1$ et $z \in \Sigma^* 2$. Dans ce cas le relateur de base est sous la forme :

$$0^n z \ \cong \ 10^n.$$

On considère le langage

$$C = 0 \ \cup \ (\bigcup_{i=0}^{n-1} 10^i)^* (\Sigma - \{0, 1\}).$$

On va montrer que $\widetilde{C}$ est un ω-code générateur de L^ω.

On a

$$\Sigma^\omega - C\Sigma^\omega = (\bigcup_{i=0}^{n-1} 10^i)^\omega \ \cup \ (\bigcup_{i=0}^{n-1} 10^i)^* 10^n \Sigma^\omega.$$

Pour montrer que $\widetilde{C}$ est un générateur de L^ω, on montre d'abord l'assertion : pour tout $x \in \Sigma^\omega - C\Sigma^\omega$, il existe $y \in C\Sigma^\omega$ tel que $x \ \cong \ y$. En effet, pour $x \in (\bigcup_{i=0}^{n-1} 10^i)^\omega$, on peut écrire x sous la forme

$$x = 10^{i_1} 10^{i_2} \ldots$$

où $n > i_j \geq 0$, on a donc

$$10^{i_1} 10^{i_2} \ldots \ \cong \ 0^n z 0^{i_1} z 0^{i_2} \ldots \in C\Sigma^\omega.$$

Pour $x \in (\bigcup_{i=0}^{n-1} 10^i)^* 10^n \Sigma^\omega$, on peut écrire x sous la forme

$$x = 10^{i_1} \dots 10^{i_p} 10^n w$$

où $p \geq 0$, $n > i_j \geq 0$ et $w \in \Sigma^\omega$, on a donc

$$10^{i_1} \dots 10^{i_p} 10^n w \ \cong \ 0^n z 0^{i_1} z \dots z 0^{i_p} w \in C\Sigma^\omega$$

ce qui achève la preuve de l'assertion. Alors on a $L^\omega \subseteq \widetilde{C} L^\omega$. D'après le lemme 4.2.12, on a $L^\omega \subseteq \widetilde{C}^\omega$. Ainsi $\widetilde{C}^\omega = L^\omega$.

Montrons à présent que $\widetilde{C}$ est un ω-code. En effet, il est clair que C est un code préfixe. De plus, on observe que

$$10^n \notin \mathrm{Fact}(C^\omega) \qquad \text{et} \qquad (\bigcup_{i=0}^{n-1} 10^i)^\omega \cap \mathrm{Suff}(C^\omega) = \emptyset,$$

on a donc $R(L) \cap (C^\omega \times C^\omega) = \emptyset$, d'après le lemme 3.1.3, $\widetilde{C}$ est un ω-code.

Exemple 14 (reprise de p. 17). Soient $L = \{a, ab, ba\}$ et $\Sigma = \{0, 1, 2\}$. L est un langage à un relateur $02 \ \cong \ 10$. L^ω a un ω-code générateur $\widetilde{C}$ avec $C = 0 \cup 1^*2$.

Cas (ii) : $k = 1, m \geq 1, n \geq 1$ et $z = 2$. Dans ce cas le relateur de base est sous la forme :

$$0^n 2 \ \cong \ 10^m 0^n.$$

On considère le langage

$$C = \{0, 2\} \cup \left(\bigcup_{i=0}^{n-1} 10^m 0^i \right)^* \left(\bigcup_{i=0}^{m-1} 10^i (\Sigma - \{0, 2\}) \cup \bigcup_{i=0}^{n-1} 10^i 2 \ \cup \Sigma - \{0, 1, 2\} \right).$$

On va montrer que $\widetilde{C}$ est un ω-code générateur de L^ω.

On a

$$\Sigma^\omega - C\Sigma^\omega = \left(\bigcup_{i=0}^{n-1} 10^m 0^i \right)^\omega \ \cup \ \left(\bigcup_{i=0}^{n-1} 10^m 0^i \right)^* (10^m 0^n \cup 10^n 0^* 2) \Sigma^\omega.$$

Pour montrer que $\widetilde{C}$ est un générateur de L^ω, on montre d'abord l'assertion : pour tout $x \in \Sigma^\omega - C\Sigma^\omega$, il existe $y \in C\Sigma^\omega$ tel que $x \cong y$. En effet, pour $x \in \left(\bigcup_{i=0}^{n-1} 10^m0^i\right)^\omega$, on peut écrire x sous la forme

$$x = 10^m0^{i_1}10^m0^{i_2}\dots$$

où $n > i_j \geq 0$, on a donc

$$10^m0^{i_1}10^m0^{i_2}\dots \cong 0^n20^{i_1}20^{i_2}\dots \in C\Sigma^\omega.$$

Pour $x \in \left(\bigcup_{i=0}^{n-1} 10^m0^i\right)^* 10^m0^n\Sigma^\omega$, on peut écrire x sous la forme

$$x = 10^m0^{i_1}\dots10^m0^{i_p}10^m0^nw$$

avec $p \geq 0$, $n > i_j \geq 0$ et $w \in \Sigma^\omega$, on a

$$10^m0^{i_1}\dots10^m0^{i_p}10^m0^nw \cong 0^n20^{i_1}2\dots20^{i_p}w \in C\Sigma^\omega.$$

Pour $x \in \left(\bigcup_{i=0}^{n-1} 10^m0^i\right)^* 10^n0^*2\Sigma^\omega$, on peut écrire x sous la forme

$$x = 10^m0^{i_1}\dots10^m0^{i_p}10^n0^q2w$$

avec $q \geq 0$, $p \geq 0$, $n > i_j \geq 0$ et $w \in \Sigma^\omega$. Si $q \geq m$, on a

$$x = 10^m0^{i_1}\dots10^m0^{i_p}10^n0^m0^{q-m}2w \cong 0^n20^{i_1}2\dots20^{i_p}20^{q-m}2w \in C\Sigma^\omega,$$

si $q < m$, on a

$$\begin{aligned} 10^m0^{i_1}\dots10^m0^{i_p}(10^n0^q)2w &= 10^m0^{i_1}\dots10^m0^{i_p}(10^q0^n)2w \\ &\cong 10^m0^{i_1}\dots10^m0^{i_p}10^q(10^m0^n)w \\ &\in (10^m0^{i_1}\dots10^m0^{i_p})10^q1\Sigma^\omega \subseteq C\Sigma^\omega \end{aligned}$$

ce qui achève la preuve de l'assertion. Alors $L^\omega \subseteq \widetilde{C}L^\omega$. D'après le lemme 4.2.12, on a

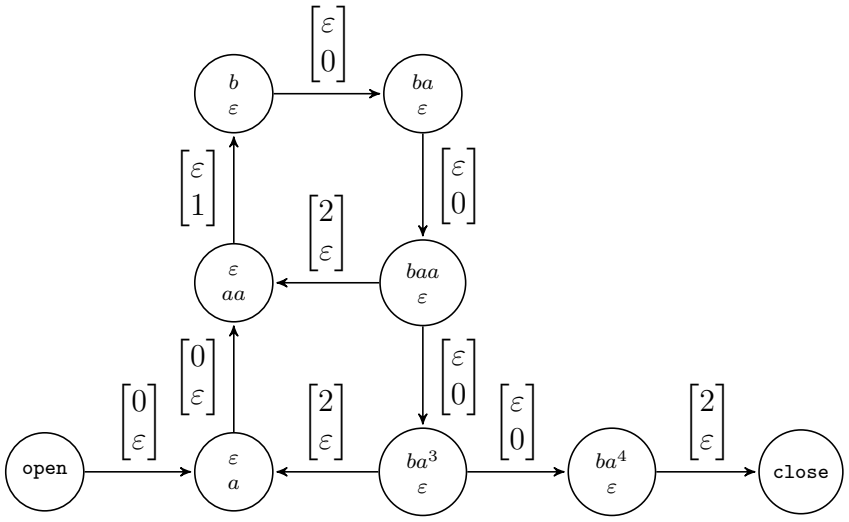


FIGURE 4.5 – Graphe de domino du langage $\{a, a^2b, ba^4\}$.

$L^\omega \subseteq \widetilde{C}^\omega$. Donc $\widetilde{C}^\omega = L^\omega$.

On observe que C est un code préfixe. De plus, on a

$$\left(\bigcup_{i=0}^{n-1} 10^m 0^i\right)^\omega \cap \mathrm{Suff}(C^\omega) = \emptyset,$$

et s'il existe $u \in \Sigma^*$ et $w \in \Sigma^\omega$ tels que $u10^m0^nw \in C^\omega$, alors $u1 \in C^+$. Ainsi on a $R(L) \cap (C^\omega \times C^\omega) = \emptyset$. D'après le lemme 3.1.3, $\widetilde{C}$ est un ω-code.

Exemple 23. Soient $L = \{a, a^2b, ba^4\}$ et $\Sigma = \{0, 1, 2\}$. D'après le graphe de domino de la figure 4.5, L est un langage à un relateur $0^22 \cong 10^20^2$. Alors L^ω a un ω-code générateur $\widetilde{C}$ correspondant à

$$C = \{0, 2\} \cup \{10^2, 10^20\}^*\{11, 101, 12, 102\}.$$

Cas (iii) : $k > 1, m = 0, n = 1$ et $z \in \Sigma^*2$. Dans ce cas le relateur de base est sous la forme :

$$0z \cong 1^k0.$$

On considère le langage

$$C = \bigcup_{i=0}^{k-1} 1^i 0 \ \cup \ 1^* \left(\Sigma - \{0,1\}\right).$$

On va montrer que $\widetilde{C}$ est un code générateur de L^ω.

On a

$$\Sigma^\omega - C\Sigma^\omega = 1^\omega \ \cup \ 1^* 1^k 0 \Sigma^\omega.$$

On montre d'abord l'assertion : pour tout $x \in \Sigma^\omega - C\Sigma^\omega$, il existe $y \in C\Sigma^\omega$ tel que $x \cong y$. En effet, pour $x = 1^\omega$, on a

$$1^\omega \cong 0z^\omega \in C\Sigma^\omega,$$

et pour $x \in 1^* 1^k 0 \Sigma^\omega$, on peut écrire x sous la forme

$$x = 1^q 1^{pk} 0 w \qquad \text{avec} \quad p \geq 0, k > q \geq 0 \text{ et } w \in \Sigma^\omega$$

on a donc

$$1^q 1^{pk} 0 w \cong 1^q 0 z^p w \in C\Sigma^\omega.$$

ce qui achève la preuve de l'assertion. On a donc $L^\omega \subseteq \widetilde{C} L^\omega$. D'après le lemme 4.2.12, on a $L^\omega \subseteq \widetilde{C}^\omega$. Ainsi $\widetilde{C}^\omega = L^\omega$.

On observe que C est un code préfixe. De plus, comme $1^k 0 \notin \mathrm{Fact}(C^+)$, on obtient $R(L) \cap (C^+ \times C^+) = \emptyset$. D'après le lemme 3.1.3, $\widetilde{C}$ est un code.

Cependant, $\widetilde{C}$ n'est pas un ω-code car

$$10z^\omega \cong 0z^\omega,$$

et $\{0, 10\} \subseteq C$.

Exemple 17 (reprise de p. 32). Soient $L = \{a, ab, baba\}$ et $\Sigma = \{0, 1, 2\}$. D'après le graphe de domino dans la figure 3.2, L est un langage à un relateur $02 \cong 110$. Alors L^ω a un code non ω-code générateur $\widetilde{C}$ avec $C = \{0, 10\} \cup 1^*2$.

Cas (iv) : $k > 1, m \geq 1, n = 1$ et $z = 2$. Dans ce cas le relateur de base est

$$02 \ \cong \ (10^m)^k 0.$$

On considère le langage

$$C = 2 \ \cup \ \bigcup_{i=0}^{k-1} (10^m)^i 0 \ \cup \ (10^m)^* \left(12 \ \cup \ 1 \left(\bigcup_{i=0}^{m-1} 0^i \right) (\Sigma - \{0, 2\}) \ \cup \ \Sigma - \{0, 1, 2\} \right).$$

On va montrer que $\widetilde{C}$ est un code générateur de L^ω.

On a

$$\Sigma^\omega - C\Sigma^\omega = (10^m)^\omega \ \cup \ (10^m)^* \left((10^m)^k 0 \cup 10^+ 2 \right) \Sigma^\omega.$$

On montre d'abord l'assertion : pour tout $x \in \Sigma^\omega - C\Sigma^\omega$, il existe $y \in C\Sigma^\omega$ tel que $x \ \cong \ y$. En effet, pour $x = (10^m)^\omega$, on a

$$(10^m)^\omega \ \cong \ 02^\omega \in C\Sigma^\omega.$$

Pour $x \in (10^m)^*(10^m)^k 0\Sigma^\omega$, on peut écrire x sous la forme

$$x = (10^m)^q (10^m)^{pk} 0w$$

avec $p \geq 0, k > q \geq 0$ et $w \in \Sigma^\omega$, on a donc

$$(10^m)^q (10^m)^{pk} 0w' \ \cong \ (10^m)^q 02^p 0w' \in C\Sigma^\omega.$$

Pour $x \in (10^m)^* 10^+ 2\Sigma^\omega$, on peut écrire x sous la forme

$$x = (10^m)^q 10^p 02w$$

avec $q \geq 0, m > p \geq 0$ et $w \in \Sigma^\omega$. On a donc

$$(10^m)^q 10^p 02w' \ \cong \ (10^m)^q 10^p (10^m)^k 0w' \in (10^m)^q 10^p 1\Sigma^\omega \subseteq C\Sigma^\omega$$

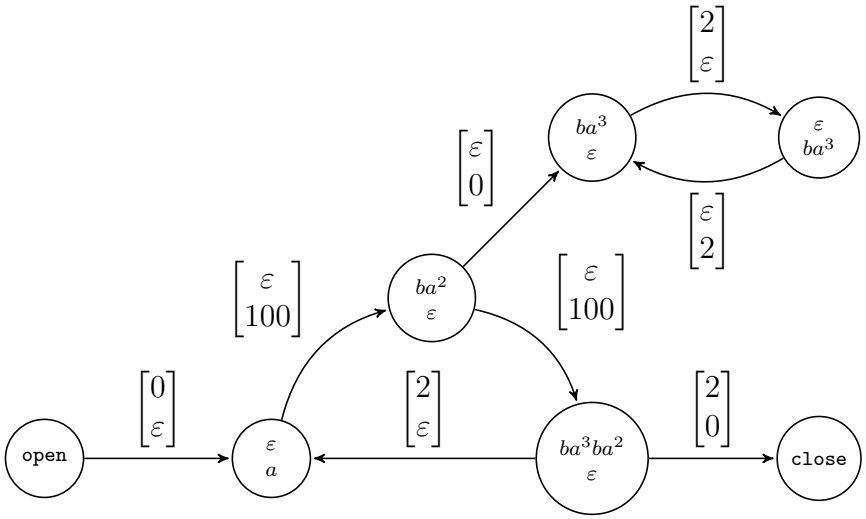


FIGURE 4.6 – Graphe de domino du langage $\{a, ab, ba^3ba^3\}$.

ce qui achève la preuve de l'assertion. Alors $L^\omega \subseteq \widetilde{C}L^\omega$. D'après le lemme 4.2.12, on a $L^\omega \subseteq \widetilde{C}^\omega$. Ainsi $\widetilde{C}^\omega = L^\omega$.

On observe que C est un code préfixe. De plus, comme $(10^m)^k 0 \notin \mathrm{Fact}(C^+)$, on obtient $R(L) \cap (C^+ \times C^+) = \emptyset$. D'après le lemme 3.1.3, $\widetilde{C}$ est un code.

Cependant $\widetilde{C}$ n'est pas un ω-code car

$$10^m 02^\omega \;\cong\; 02^\omega$$

et $\{0, 10^m 0, 2\} \subseteq C$.

Exemple 24. Soient $L = \{a, ab, ba^3ba^3\}$ et $\Sigma = \{0, 1, 2\}$. D'après le graphe de domino de la figure 4.6, L est un langage à un relateur $02 \;\cong\; (100)^2 0$. Alors L^ω a un code générateur $\widetilde{C}$ avec

$$C = \{0, 1000, 2\} \cup (100)^*\{12, 11, 101\}.$$

On vérifie que $\widetilde{C}$ n'est pas un ω-code car

$$aba^3(ba^3ba^3)^\omega = a(ba^3ba^3)^\omega.$$

La deuxième partie de théorème 4.1.7 se déduit de la propriété suivante.

Propriété 4.3.8. *Si le relateur de base de L est de la forme (iii) ou (iv), alors L^ω n'a pas d'ω-code générateur.*

4.3.1 Démonstration de la propriété 4.3.1

On rappelle la propriété que l'on montre dans cette section :

Propriété 4.3.1. *Si* $\mathrm{OVL}(u,v) = \emptyset$ *et* $\mathrm{OVL}(v,u) = \emptyset$, *alors L^ω n'a pas de code générateur.*

Du fait que $\mathrm{OVL}(u,v) = \mathrm{OVL}(v,u) = \emptyset$, on a $\mathrm{LB}(u,v) = \mathrm{LB}(v,u) = \emptyset$. D'après le lemme 4.2.2, on a

$$Amb_\Sigma(L) = \Sigma^*\{u,v\}\Sigma^\omega \cup \Sigma^* \left(\mathrm{LB}(u,u) \cup \mathrm{LB}(v,v)\right)^\omega . \tag{4.4}$$

Comme u et v jouent un rôle symétrique, on a trois cas :

Cas 1 : $u \in \{0,1\}^+$ et $v \in \{0,1\}^+$. D'après le lemme 4.2.8, les deux mots $\tilde{0}$ et $\tilde{1}$ commutent. Ceci est en contradiction avec le résultat du lemme 4.2.6.

Cas 2 : $u \notin \{0,1\}^+$ et $v \notin \{0,1\}^+$. D'après le fait 4.2.3, on a

$$\mathrm{LB}(u,u) \cap \{0,1\}^+ = \emptyset \quad \text{et} \quad \mathrm{LB}(v,v) \cap \{0,1\}^+ = \emptyset.$$

D'après (4.4), on a $Amb_\Sigma(L) \cap \{0,1\}^\omega = \emptyset$. D'après la proposition 3.2.6, L^ω n'a pas de code générateur.

Cas 3 : $u \in \{0,1\}^+$ et $v \notin \{0,1\}^+$. D'après (4.4) on a

$$\begin{aligned} Amb_\Sigma(L) \cap \{0,1\}^\omega &= \{0,1\}^* u \{0,1\}^\omega \cup \{0,1\}^* \left(\mathrm{LB}(u,u)\right)^\omega \\ &= \{0,1\}^* u \{0,1\}^\omega \qquad \text{(d'après le fait 4.2.4)} \end{aligned}$$

Comme $\mathrm{First}(v) = 1$ et $\mathrm{OVL}(v,u) = \emptyset$, on a $\mathrm{Last}(u) \neq 1$. Donc $\mathrm{Last}(u) = 0$ et $0 \in \mathrm{OVL}(u,u)$. On peut écrire $u = 0z0$ où $z \in \{0,1\}^*$. Maintenant, le relateur de base

devient $0z0 \cong v$. Comme

$$0zv \;\cong\; 0z(0z0) = (0z0)0z \;\cong\; vz0,$$

on a $0zv \cong vz0$ où $\mathrm{First}(v) = 1$. Ainsi les deux mots $0z1$ et 1 sont incompatibles. De plus, d'après le lemme 4.2.11, on a

$$u = 0z0 \notin \mathrm{Fact}(\{0z1, 1\}^\omega).$$

Donc $Amb_\Sigma(L) \cap \{0z1, 1\}^\omega = \emptyset$. D'après la proposition 3.2.6, L^ω n'a pas de code générateur.

Exemple 15 (pour illustrer le cas 2, reprise de p. 25). Soient $L = \{a, ab, bc, c\}$ et $\Sigma = \{0, 1, 2, 3\}$. L est un langage à un relateur $02 \;\cong\; 13$.

Exemple 20 (pour illustrer le cas 3, reprise de p. 39). Soient $L = \{ab, aba, b\}$ et $\Sigma = \{0, 1, 2\}$. L est un langage à un relateur $00 \;\cong\; 12$.

4.3.2 Démonstration de la propriété 4.3.2

On rappelle la propriété que l'on montre dans cette section :

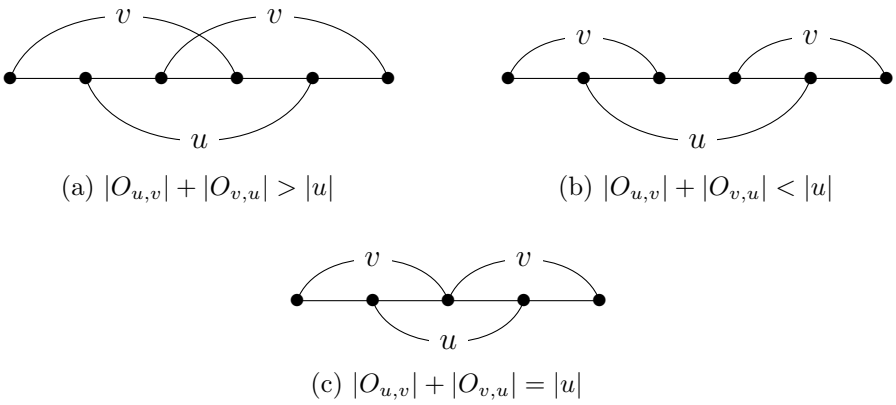
Propriété 4.3.2. *Si* $\mathrm{OVL}(u, v) \neq \emptyset$ *et* $\mathrm{OVL}(v, u) \neq \emptyset$*, alors* L^ω *n'a pas de code générateur.*

On montre d'abord que

$$|u| > |O_{u,v}| + |O_{v,u}| \quad \text{et} \quad |v| > |O_{u,v}| + |O_{v,u}|.$$

Supposons au contraire que $|u| \leq |O_{u,v}| + |O_{v,u}|$. Comme $|\tilde{u}| = |\tilde{v}|$, on doit avoir $|v| \leq |O_{u,v}| + |O_{v,u}|$. Alors le relateur de base $u \cong v$ peut s'écrire sous la forme

$$u_1u_2u_3 \;\cong\; v_1v_2v_3,$$



(a) $|O_{u,v}| + |O_{v,u}| > |u|$ (b) $|O_{u,v}| + |O_{v,u}| < |u|$

(c) $|O_{u,v}| + |O_{v,u}| = |u|$

FIGURE 4.7 – Situations où deux mots u et v chevauchent des deux côtés.

où $u_i, v_i \in \Sigma^*$, $u_1u_2 = v_2v_3 = O_{u,v}$ et $u_2u_3 = v_1v_2 = O_{v,u}$. On obtient donc

$$u_1v_1v_2 \;\cong\; v_1u_1u_2.$$

Comme $|\widetilde{u_1v_1}| = |\widetilde{v_1u_1}|$, on a $u_1v_1 \cong v_1u_1$ et $v_2 \cong u_2$. De plus, comme $u_1v_1v_2 \cong v_1u_1u_2$ est un relateur minimal, on doit avoir $u_2 = v_2 = \varepsilon$ ou $u_1v_1 = v_1u_1 = \varepsilon$. Si $u_2 = v_2 = \varepsilon$ alors le relateur de base est $u_1v_1 \cong v_1u_1$; ceci est en contradiction avec le résultat du lemme 4.2.6. Si $u_1v_1 = v_1u_1 = \varepsilon$, alors $O_{u,v} = u_2 = v_2v_3$ et $O_{v,u} = v_2 = u_2u_3$ d'où $u_3 = v_3 = \varepsilon$ et donc $O_{u,v} = u$; ceci est en contradiction avec la définition de $\mathrm{OVL}(u, v)$.

Rappelons que

$$\mathrm{First}(O_{u,v}) = \mathrm{First}(u) = 0 \neq 1 = \mathrm{First}(v) = \mathrm{First}(O_{v,u}).$$

Maintenant on écrit le relateur de base sous la forme

$$0xz1t \cong\; 1ty0x$$

où $x, t \in \Sigma^*$; $y, z \in \Sigma^+$; $0x = O_{u,v}$ et $1t = O_{v,u}$. Comme $0xz$ et $1ty$ jouent un rôle symétrique, on considère trois cas :

Cas 1 : $0xz \in \{0,1\}^+$ et $1ty \in \{0,1\}^+$. D'après le lemme 4.2.8, les deux mots $\tilde{0}$ et $\tilde{1}$

commutent. C'est une contradiction avec le corollaire 4.2.7.

Cas 2 : $0xz \notin \{0,1\}^+$ et $1ty \notin \{0,1\}^+$. D'après le lemme 4.2.2, on a donc

$$Amb_\Sigma(L) \cap \{0,1\}^\omega = \emptyset.$$

D'après la proposition 3.2.6, L^ω n'a pas de code générateur.

Cas 3 : $0xz \in \{0,1\}^+$ et $1ty \notin \{0,1\}^+$. On montre d'abord $\mathrm{Last}(1t) \notin \{0,1\}$. En effet, si $\mathrm{Last}(1t) = 0$ alors (d'après le lemme 4.2.8) les deux mots $\tilde{0}$ et $\tilde{1}$ commutent ; si $\mathrm{Last}(1t) = 1$ alors $\mathrm{Last}(0x) = 0$, d'après le lemme 4.2.9, les deux mots $\tilde{0}$ et $\tilde{1}$ commutent. Ces deux cas conduisent à une contradiction avec le corollaire 4.2.7.

Maintenant on a $\mathrm{Last}(1t) \notin \{0,1\}$. Donc

$$\mathrm{LB}(u,u) \cap \{0,1\}^+ = \emptyset, \quad \mathrm{LB}(v,v) \cap \{0,1\}^+ = \emptyset, \quad \mathrm{LB}(u,v) \cap \{0,1\}^+ = \emptyset$$

et d'après le fait 4.2.5, on a $\mathrm{LB}(v,u) \cap \{0,1\}^+ = 0xz$. Donc on a

$$Amb_\Sigma(L) \cap \{0,1\}^\omega = \{0,1\}^*(0xz)^\omega.$$

D'après le lemme 4.2.11, on a $0xz0 \notin \mathrm{Fact}(\{0xz1,1\}^\omega)$. Alors

$$Amb_\Sigma(L) \cap \{0xz1,1\}^\omega = \emptyset.$$

Comme les deux mots $0xz1$ et 1 sont incompatibles, d'après la proposition 3.2.6, L^ω n'a pas de code générateur.

Exemple 25 (pour illustrer le cas 2). Soient $L = \{a, aba, bac, cab\}$ et $\Sigma = \{0,1,2,3\}$. D'après le graphe de domino de la figure 4.8, L est un langage à un relateur $021 \cong 130$.

Je ne trouve pas d'exemple pour illustrer le cas 3 mais je pense qu'il en existe. Aussi, je propose de laisser la recherche d'un tel exemple à l'état de question.

Question. Existe-il un langage à un relateur $0xz1t \cong 1ty0x$ où $xz \in \{0,1\}^+$, $ty \notin \{0,1\}^+$ et $\mathrm{Last}(1t) \notin \{0,1\}$?

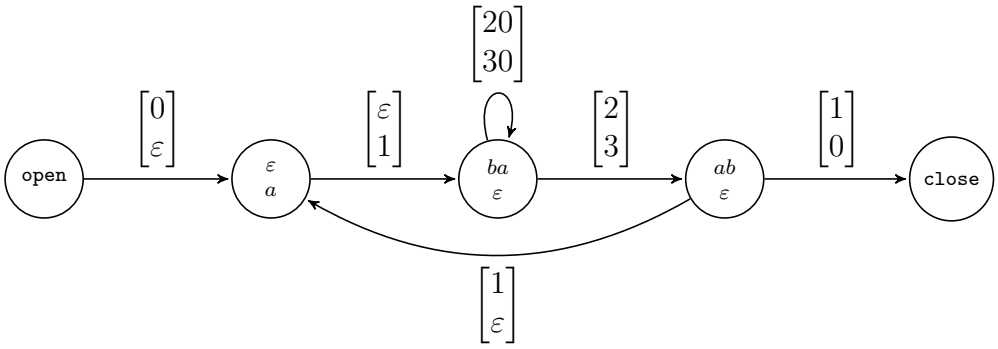


FIGURE 4.8 – Graphe de domino (qui est simplifié en regroupant des arêtes) du langage $\{a, aba, bac, cab\}$.

4.3.3 Démonstration de la propriété 4.3.5

On rappelle la notation suivante :

Notation 4.3.3. *On réécrit le relateur de base $u \cong v$ sous la forme*

$$0xz \;\cong\; 1y0x \tag{4.2}$$

où $z \in \Sigma^+$; $x, y \in \Sigma^*$; et $0x = O_{u,v}$ est le plus grand élément (pour la longueur) de $\mathrm{OVL}(u, v)$.

La propriété que l'on montre dans cette section est :

Propriété 4.3.5. *Soit le relateur de base de L sous la forme* (4.2)*. Si $x \notin 0^*$ ou $\sqrt{1y} \notin 10^*$, alors L^ω n'a pas de code générateur.*

Supposons que $x \notin 0^*$ ou $\sqrt{1y} \notin 10^*$. Alors on considère les cas suivants :

Cas 1 : $y \notin \{0,1\}^*$. D'après le fait 4.3.4, on a $xz \notin \{0,1\}^+$. Donc

$$Amb_\Sigma(L) \cap \{0,1\}^\omega = \emptyset.$$

D'après la proposition 3.2.7, alors L^ω n'a pas de code générateur.

Exemple 26 (pour illustrer le cas 1). Soient $L = \{a, ab, bca, c\}$ et $\Sigma = \{0, 1, 2, 3\}$. D'après le graphe de domino de la figure 4.9, L est un langage à un relateur $02 \;\cong\; 130$.

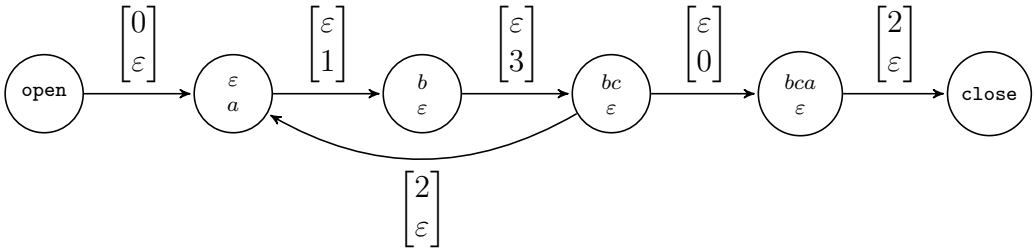


FIGURE 4.9 – Graphe de domino du langage $\{a, ab, bca, c\}$.

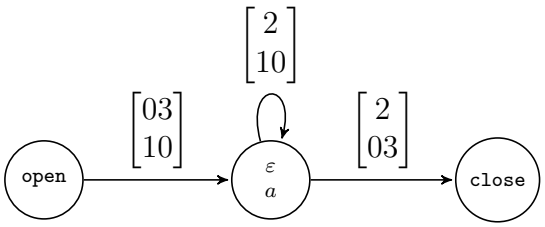


FIGURE 4.10 – Graphe simplifié de domino du langage $\{ab, abc, bcaba, caba\}$.

Cas 2 : $y \in \{0,1\}^*$ et $x \notin 0^*$. Comme le relateur de base est $0xz \cong 1y0x$, si $x \in \{0,1\}^+ - 0^+$, d'après le lemme 4.2.8, les deux mots $\tilde{0}$ et $\tilde{1}$ commutent. Ceci est en contradiction avec le fait que L est un langage à un relateur. Donc on a $x \notin \{0,1\}^+$. D'après le lemme 4.2.2, on déduit

$$Amb_\Sigma(L) \cap \{0,1\}^\omega = \{0,1\}^*(1y)^\omega.$$

D'après le lemme 4.2.11, on a donc $1y1 \notin \text{Fact}(\{0, 1y0\}^\omega)$. Donc

$$Amb_\Sigma(L) \cap \ \{0, 1y0\}^\omega = \emptyset.$$

De plus, les deux mots 0 et $1y0$ sont incompatibles car $0xz \cong 1y0x$. Donc d'après la proposition 3.2.7, L^ω n'a pas de code générateur.

Exemple 27 (pour illustrer le cas 2). Soient $L = \{ab, abc, bcaba, caba\}$ et $\Sigma = \{0, 1, 2, 3\}$. D'après le graphe simplifié de domino de la figure 4.10, L est un langage à un relateur $032 \cong 1003$.

Cas 3 : $0x = 0^n, y \in \{0,1\}^*$, $\sqrt{1y} \notin 10^*$ avec $n > 0$. Comme $xz \notin \{0,1\}^*$ on peut écrire $z = z_{01}2z'$ avec $z_{01} \in \{0,1\}^*$ et $z' \in \Sigma^*$. Maintenant le relateur de base est de la forme $0^n z_{01} 2z' \cong 1y0^n$ avec $y \in \{0,1\}^*$ et $z_{01} \in \{0,1\}^*$. D'après le lemme 4.2.6 et le lemme 4.2.8, on en déduit alors qu'il existe un entier $k \geq 0$ tel que $z_{01} = 0^k$. C'est-à-dire, le relateur de base peut s'écrire sous la forme

$$0^n 0^k 2z' \ \cong \ 1y0^n. \tag{4.5}$$

D'après le lemme 4.2.2, on a

$$Amb_\Sigma(L) \cap \{0,1\}^\omega = \{0,1\}^* 1y0^n \{0,1\}^\omega \cup \{0,1\}^* \left(\mathrm{LB}(0^n 0^k 2z', 1y0^n) \cup \mathrm{LB}(1y0^n, 1y0^n)\right)^\omega . \tag{4.6}$$

Il y a deux sous-cas :

Sous-cas 3.1 : $\mathrm{OVL}(1y0^n, 1y0^n) \neq \emptyset$. Donc il existe $f \in \{0,1\}^*$ et $p \in \{0,1\}^*$ tels que $1y0^n = 1pf1p$. On remarque que $1pf1 \in \mathrm{Pref}(1y)$. Le relateur de base (4.5) peut être écrit sous la forme

$$0^n 0^k 2z' \ \cong \ 1pf1p.$$

On a

$$0^n 0^k 2z' f1p \ \cong \ 1pf1pf1p \ \cong \ 1pf0^n 0^k 2z',$$

donc

$$0^n 0^k 2z' f1p \ \cong \ 1pf0^n 0^k 2z'.$$

Alors les deux mots 0 et $1pf0$ sont incompatibles. De plus, comme $1pf1 \notin \mathrm{Fact}(\{0, 1pf0\}^\omega)$ (d'après le lemme 4.2.11) ; donc $1y \notin \mathrm{Fact}(\{0, 1pf0\}^\omega)$. Donc on a

$$Amb_\Sigma(L) \cap \{0, 1pf0\}^\omega = \emptyset$$

D'après la proposition 3.2.7, L^ω n'a pas de code générateur.

Sous-cas 3.2 : $\mathrm{OVL}(1y0^n, 1y0^n) = \emptyset$. À partir de l'équation (4.6), on a

$$Amb_\Sigma(L) \cap \{0,1\}^\omega = \{0,1\}^* 1y0^n \{0,1\}^\omega \cup \{0,1\}^* \{1y, 1y0, \cdots, 1y0^{n-1}\}^\omega.$$

À partir du relateur de base (4.5), on a

$$0^n 0^k 2z' 0^k 2z' \;\cong\; 1y0^n 0^k 2z' \;\cong\; 1y1y0^n.$$

Les deux mots 0 et $1y1$ sont donc incompatibles. On va montrer que

$$Amb_\Sigma(L) \cap \{0, 1y1\}^\omega = \emptyset \tag{4.7}$$

et d'après la proposition 3.2.7, L^ω n'a pas de code générateur.

Pour montrer l'assertion (4.7), il suffit de montrer que

$$1y0^n \notin \mathrm{Fact}(\{0, 1y1\}^\omega) \tag{4.8}$$

et

$$\left(\bigcup_{i=0}^{n-1} 1y0^i\right)^\omega \cap \mathrm{Suff}(\{0, 1y1\}^\omega) = \emptyset. \tag{4.9}$$

Si $1y0^n \in \mathrm{Fact}(\{0, 1y1\}^\omega)$ alors il existe un entier $m \geq 0, p \in \mathrm{Pref}(1y) - 1y$ et $s \in \mathrm{Suff}(1y)$ tels que

$$1y0^n = s10^m p.$$

On a $p = \varepsilon$ car $\mathrm{OVL}(1y0^n, 1y0^n) = \emptyset$. Donc

$$1y0^n \in \mathrm{Suff}(1y)10^m.$$

On déduit donc que $\sqrt{1y} \in 10^*$: contradiction. Cela montre l'assertion (4.8).

Si on a

$$\left(\bigcup_{i=0}^{n-1} 1y0^i\right)^\omega \cap \mathrm{Suff}(\{0, 1y1\}^\omega) = \emptyset$$

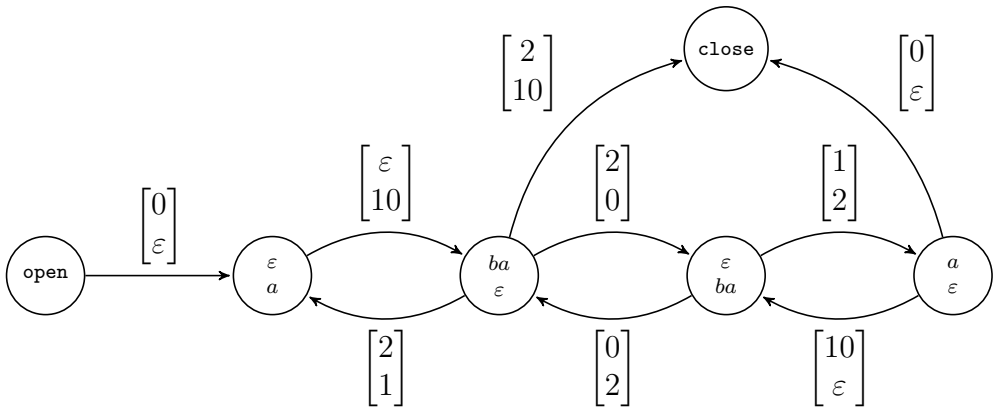


FIGURE 4.11 – Graphe simplifié de domino du langage $\{a, ab, baaba\}$.

alors il existe deux suites $(i_j), (k_j)$ et un mot $s \in (\mathrm{Suff}(1y1) - 1y1)$ tels que

$$1y0^{i_1}1y0^{i_2}\ldots = s0^{k_1}1y10^{k_2}\ldots.$$

On pose $\ell = |1y|_1 - |s|_1 \geq 0$. Comme

$$\begin{aligned}|s|_1 + |(1y1)^\ell|_1 &= |s|_1 + \ell + |(1y)^\ell|_1 \\ &= |1y|_1 + |(1y)^\ell|_1 = |(1y)^{\ell+1}|_1\end{aligned}$$

D'après le fait que : si $\alpha 1$ et $\beta 1$ sont des préfixes d'un même mot et si $|\alpha|_1 = |\beta|_1$ alors $\alpha = \beta$, on a

$$1y0^{i_1}\ldots 1y0^{i_{\ell+1}} = s0^{k_1}1y10^{k_2}\ldots 1y10^{k_\ell}.$$

Donc

$$1y0^{i_{\ell+1}} \in \mathrm{Suff}(1y)10^{k_\ell}.$$

On déduit donc $\sqrt{1y} \in 10^*$: contradiction. Cela montre l'assertion (4.9).

Exemple 28 (pour illustrer le sous-cas 3.1). Soient $L = \{a, ab, baaba\}$ et $\Sigma = \{0, 1, 2\}$. D'après le graphe simplifié de domino de la figure 4.11, on peut vérifier comme dans l'exemple 20 que L est un langage à un relateur $02 \cong 1010$.

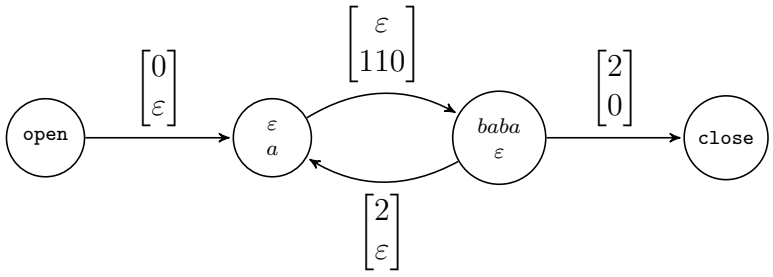


FIGURE 4.12 – Graphe simplifié de domino du langage $\{a, ab, babaa\}$.

Exemple 29 (pour illustrer le sous-cas 3.2). Soient $L = \{a, ab, babaa\}$ et $\Sigma = \{0, 1, 2\}$. D'après le graphe simplifié de domino de la figure 4.12, L est un langage à un relateur $02 \cong 1100$.

4.3.4 Démonstration de la propriété 4.3.6

On rappelle la propriété que l'on montre dans cette section :

Propriété 4.3.6. *Soit le relateur de base de L sous la forme*

$$0^n z \cong (10^m)^k 0^n \quad \textit{avec} \quad 0^n = O_{u,v} \textit{ et } \mathrm{Last}(z) \notin \{0, 1\}.$$

Si $k \geq 2, m \geq 0$ et $n \geq 2$, alors L^ω n'a pas de code générateur.

D'après le lemme 4.2.2, on a

$$Amb_\Sigma(L) \cap \{0,1\}^\omega = \{0,1\}^*(10^m)^k 0^n \{0,1\}^\omega \cup \{0,1\}^* \left(\bigcup_{i=0}^{n-1} (10^m)^k 0^i \right)^\omega$$

À partir du relateur de base $0^n z \cong (10^m)^k 0^n$, on déduit

$$0^n z 0 z \cong (10^m)^k 0^n 0 z \cong (10^m)^k 0 (10^m)^k 0^n.$$

Ainsi les deux mots 0 et $(10^m)^k 01$ sont incompatibles.

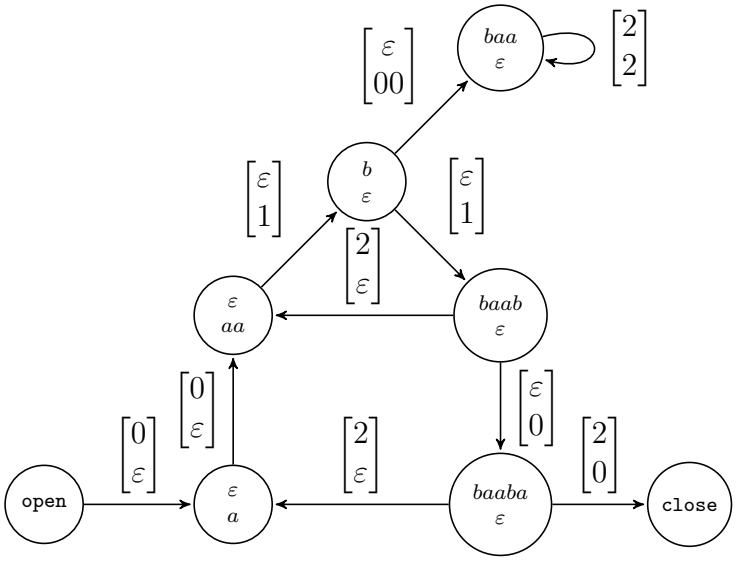


FIGURE 4.13 – Graphe simplifié de domino du langage $\{a, aab, baabaa\}$.

Comme $k \geq 2$ et $n \geq 2$, on a

$$(10^m)^k 0^n \notin \text{Fact}(\{0, (10^m)^k 01\}^\omega).$$

De plus, on a

$$\text{Suff}(\{0, (10^m)^k 01\}^\omega) \ \cap \ \left(\bigcup_{i=0}^{n-1} (10^m)^k 0^i \right)^\omega = \emptyset.$$

Alors on a

$$Amb_\Sigma(L) \ \cap \ \{0, (10^m)^k 01\}^\omega = \emptyset.$$

Donc d'après la proposition 3.2.7, L^ω n'a pas de code générateur.

Exemple 30. Soient $L = \{a, aab, baabaa\}$ et $\Sigma = \{0, 1, 2\}$. D'après le graphe simplifié de domino de la figure 4.13, L est un langage à un relateur $002 \ \cong \ 1100$.

4.3.5 Démonstration de la propriété 4.3.7

On rappelle la propriété que l'on montre dans cette section :

Propriété 4.3.7. *Soit le relateur de base de L sous la forme*

$$0^n z \;\cong\; (10^m)^k 0^n \quad \textit{avec} \quad 0^n = O_{u,v} \textit{ et } \mathrm{Last}(z) \notin \{0,1\}.$$

Si $|z| \geq 2$, $k \geq 1, m \geq 1$ et $n \geq 1$, alors L^ω n'a pas de code générateur.

On remarque que $\mathrm{First}(z) \neq 0$ car $O_{u,v} = 0^n$; d'après le lemme 4.2.8 et L ne peut pas contenir deux mots commutant, on a $\mathrm{First}(z) \neq 1$. Donc le relateur de base peut être écrit sous la forme

$$0^n 2 z' \;\cong\; (10^m)^k 0^n$$

où $z' \neq \varepsilon$ et $\mathrm{Last}(z') \notin \{0,1\}$. Donc les deux mots $0^n 2$ et 1 sont incompatibles. De plus, on a

$$\begin{aligned} &Amb_\Sigma(L) = \\ &\quad \Sigma^* \left\{0^n 2 z', (10^m)^k 0^n\right\} \Sigma^\omega \;\cup \Sigma^* \left(\mathrm{LB}(0^n 2 z', 0^n 2 z') \cup \mathrm{LB}(0^n 2 z', (10^m)^k 0^n\right)^\omega . \end{aligned} \quad (4.10)$$

Ainsi on a

$$Amb_\Sigma(L) \cap \{0^n 2, 1\}^\omega \subseteq \Sigma^* 0^n 2 z' \Sigma^\omega. \quad (4.11)$$

On considère deux cas :

(a) Si $Amb_\Sigma(L) \cap \{0^n 2, 1\}^\omega = \emptyset$. Donc d'après la proposition 3.2.7, L^ω n'a pas de code générateur.

(b) Sinon $Amb_\Sigma(L) \cap \{0^n 2, 1\}^\omega \neq \emptyset$, on doit avoir

$$0^n 2 z' \in \mathrm{Fact}(\{0^n 2, 1\}^\omega).$$

De plus, $\mathrm{Last}(z') \notin \{0,1\}$, alors $z' \in \Sigma^* 0^n 2$. Donc le relateur de base peut être écrit sous la forme

$$0^n 2 f 0^n 2 \;\cong\; (10^m)^k 0^n$$

où $f \in \mathrm{Pref}(z')$. On en déduit :

$$0^n 2 f (10^m)^k 0^n \;\cong\; 0^n 2 f 0^n 2 f 0^n 2 \cong\; (10^m)^k 0^n f 0^n 2. \quad (4.12)$$

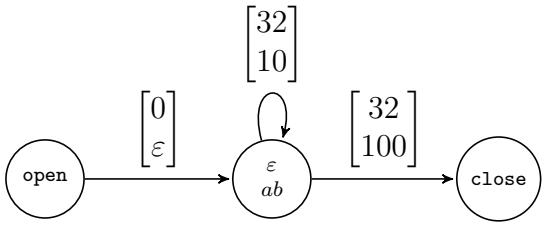


FIGURE 4.14 – Graphe simplifié de domino du langage $\{ab, abc, b, caba\}$.

Comme f ne peut pas contenir le facteur $(10^m)^k$, on a

$$Amb_\Sigma(L) \cap \{0^n 2 f 1, 1\}^\omega \subseteq \Sigma^* 0^n 2 f 0^n 2 \Sigma^\omega.$$

D'après le lemme 4.2.11, on a $0^n 2 f 0 \notin \mathrm{Fact}(\{0^n 2 f 1, 1\}^\omega)$, on a donc

$$0^n 2 f 0^n 2 \notin \mathrm{Fact}(\{0^n 2 f 1, 1\}^\omega).$$

On déduit

$$Amb_\Sigma(L) \cap \{0^n 2 f 1, 1\}^\omega = \emptyset.$$

De plus, d'après (4.12), les deux mots $0^n 2 f 1$ et 1 sont incompatibles. Donc d'après la proposition 3.2.7, L^ω n'a pas de code générateur.

Exemple 31. Soient $L = \{ab, abc, b, caba\}$ et $\Sigma = \{0, 1, 2, 3\}$. D'après le graphe simplifié de domino de la figure 4.14, L est un langage à un relateur $032 \cong 100$.

Exemple 32. Soient $L = \{a, aba^3b, ba^2\}$ et $\Sigma = \{0, 1, 2\}$. D'après le graphe simplifié de domino de la figure 4.15, on peut vérifier comme dans l'exemple 20 que L est un langage à un relateur $0202 \cong 100$.

4.3.6 Démonstration de la propriété 4.3.8

On rappelle la propriété que l'on montre dans cette section :

Propriété 4.3.8. *Si le relateur de base de L est de la forme (iii) ou (iv), alors L^ω n'a pas d'ω-code générateur.*

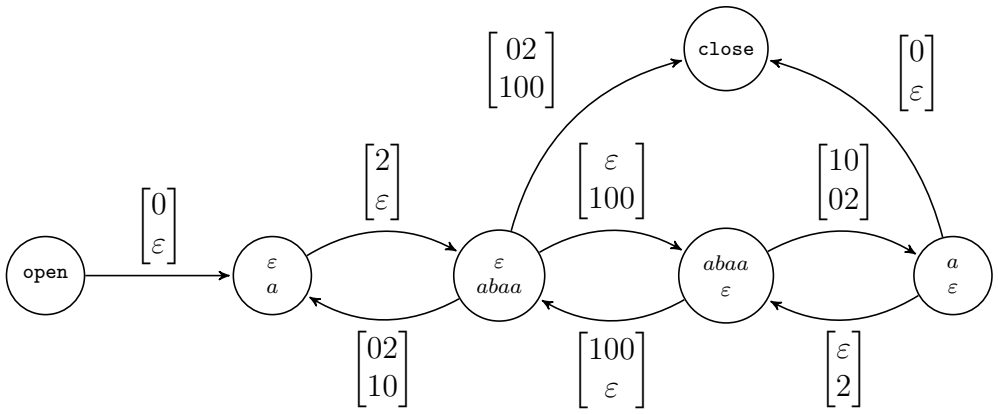


FIGURE 4.15 – Graphe simplifié de domino du langage $\{a, aba^3b, ba^2\}$.

Forme (iv) : $u = 02$ et $v = (10^m)^k 0$ où $m \geq 1$ et $k > 1$. Comme pour tout $i > 0$, on a

$$02^i \cong (10^m)^k 02^{i-1} \cong (10^m)^k (10^m)^k 02^{i-2} \cong \cdots \cong (10^m)^{ki} 0,$$

par passage à l'adhérence on a

$$02^\omega \cong (10^m)^\omega.$$

De plus,

$$10^m 02^\omega \cong 10^m (10^m)^\omega = (10^m)^\omega \cong 02^\omega.$$

Comme $k > 1$, on a

$$10^m 02^\omega \cong 02^\omega$$

est un relateur minimal. Ainsi les deux mots 0 et $10^m 0$ sont incompatibles.

On a

$$Amb_\Sigma(L) \cap \{0,1\}^\omega = \{0,1\}^*(10^m)^k 0\{0,1\}^\omega \cup \{0,1\}^*(10^m)^\omega$$

D'après le lemme 4.2.11, alors on a

$$10^m 1 \notin \mathrm{Fact}(\{10^m 0, 0\}^\omega),$$

donc

$$Amb_\Sigma(L) \cap \{10^m 0, 0\}^\omega = \emptyset.$$

D'après la proposition 3.3.5, L^ω n'a pas d'ω-code générateur.

Forme (iii) : $u = 0w2$ et $v = 1^k 0$ avec $z \in \Sigma^+$ et $k > 1$. Avec un argument comme dans la forme (iv), on montre que L^ω n'a pas d'ω-code générateur.

4.4 Démonstration du théorème 4.1.6

Supposons qu'il existe un langage $C \subseteq \Sigma^+$ tel que $\widetilde{C}$ est un code générateur de L^ω. D'après le théorème 4.1.7, on considère quatre cas :

Cas (i) : $u = 0^n w2$ et $v = 10^n$ où $w \in \Sigma^*$ et $n \geq 1$. On a

$$Amb_\Sigma(L) = \Sigma^*\{0^n w2, 10^n\}\Sigma^\omega \cup \Sigma^* \left(\mathrm{LB}(0^n w2, 0^n w2) \ \cup \ \bigcup_{i=0}^{n-1} 10^i \right)^\omega .$$

Comme $0^\omega \notin Amb_\Sigma(L)$, il existe un entier unique $i \geq 0$ tel que

$$0^i 0 \in C. \tag{4.13}$$

Comme $0^n w2 \ \cong \ 10^n$, pour tout $j > 0$, on a

$$0^n (w2)^j \ \cong \ 1^j 0^n.$$

Ainsi pour tout $j > 0$, les mots 0 et 1^j sont incompatibles. D'après (4.13) et le lemme 3.2.4, on a

$$0^i P(\{0, 1^\omega\}) \cap C = 0^i 0.$$

On a donc

$$\mathrm{Pref}(0^i 1^\omega) \cap C = \emptyset.$$

De plus, comme le relateur de base est $0^n w2 \ \cong \ 10^n$ et $\tilde{2} \neq \varepsilon$, on a $1 \notin \mathrm{Fact}(w)$. Ainsi

on a

$$0^i 1^+ 2^\omega \cap Amb_\Sigma(L) = \emptyset.$$

Alors, d'après le lemme 3.2.1, pour tout entier $p > 0$, il existe un entier $q > 0$ tel que

$$0^i 1^p 2^q \in C.$$

Alors $\widetilde{C}$ est infini.

Cas (ii) : $u = 0^n 2$ et $v = 10^m 0^n$ où $m \geq 1$ et $n \geq 1$. On a

$$Amb_\Sigma(L) = \Sigma^*\{0^n 2, 10^m 0^n\}\Sigma^\omega \cup \Sigma^*\{10^m, \ldots, 10^m 0^{n-1}\}^\omega.$$

Comme $0^\omega \notin Amb_\Sigma(L)$, il existe un entier $i \geq 0$ tel que

$$0^i 0 \in C. \tag{4.14}$$

Comme $0^n 2 \cong 10^m 0^n$, pour tout $j > 0$, on a

$$0^n 2^j \cong (10^m)^j 0^n.$$

Ainsi pour tout mot $f \in P((10^m)^\omega)$, les mots 0 et f sont incompatibles. D'après (4.14) et le lemme 3.2.4, on a

$$0^i P(\{0, (10^m)^\omega\}) \cap C = 0^i 0.$$

On a donc

$$\mathrm{Pref}(0^i (10^m)^\omega) \cap C = \emptyset.$$

De plus, on a

$$0^i (10^m)^+ 1^\omega \cap Amb_\Sigma(L) = \emptyset.$$

D'après le lemme 3.2.1, alors pour tout entier $p > 0$, il existe un entier $q > 1$ tel que

$$0^i (10^m)^p 1^q \in C.$$

Alors $\widetilde{C}$ est infini.

Cas (iii) : $u = 0w2$ et $v = 1^k0$ où $k > 1$ et $m \geq 1$. Avec un argument comme dans le cas (i), on obtient que $\widetilde{C}$ est infini.

Cas (iv) : $u = 02$ et $v = (10^m)^k0$ où $m \geq 1$ et $k > 1$. Avec un argument comme dans le cas (ii), on obtient que $\widetilde{C}$ est infini.

4.5 Remarques

On termine ce chapitre par quelques extensions et remarques. D'abord, on considère l'exemple suivant :

Exemple 20 (reprise de p. 39). Soit $L = \{a, ab, ba, ac, ca\}$. Ce langage est de la forme

$$A \cup AB \cup BA$$

avec $A = \{a\}$ et $B = \{b, c\}$. Ce langage généralise le cas du langage à un relateur $\{a, ab, ba\}$ où il y a un ω-code générateur $a \cup (ab)^*ba$. L'ω langage L^ω a un ω-code générateur

$$a \cup (a\{b, c\})^*(\{b, c\}^*a)$$

qui est de la forme

$$A \cup (AB)^*(BA).$$

Le langage dans l'exemple 20 n'est pas à un relateur mais on peut le traiter par une méthode similaire à celle des langages à un relateur. Les langages de ce type peuvent être considérés comme une extension de langages à un relateur. A et B peuvent même être des langages infinis.

L'exemple suivant est l'union de deux langages à un relateur. On peut le considérer comme un langage à deux relateurs.

Exemple 16 (reprise de p. 25). Soit $L = \{a, ab, ba, c, cd, dc\}$. Ce langage est l'union de

deux langages à un relateur qui sont similaires à celui dans l'exemple 20 :

$$L_1 = \{a, ab, ba\} \qquad \text{et} \qquad L_2 = \{c, cd, dc\}.$$

On a vu, cf. p. 25, que L^ω n'a pas de code générateur.

Le théorème 4.1.6 énonce que l'ω-puissance d'un langage à un relateur n'a pas de code fini générateur. L'exemple suivant montre que si un langage n'est pas à un relateur alors son ω-puissance peut être engendrée par des ω-codes finis.

Exemple 10 (reprise de p. 15). Soit $L = \{a^2, a^3, b\}$. Le langage fini $\{a^2, a^3b, b\}$ est un ω-code générateur de L^ω.

Chapitre 5

Langages réduits

Dans ce chapitre, nous nous intéressons à un type de générateurs intermédiaires entre les codes et les générateurs minimaux : les générateurs réduits.

5.1 Langages n-réduits

Soit n un entier et $L \subseteq A^*$ un langage, on note $L^{\leq n} = \bigcup_{i=1}^{n} L^i$.

Définition 5.1.1. Soit $n > 0$ un entier. Un langage $L \subseteq A^+$ est appellé n-réduit si pour tout $u \in L^{\leq n}$, l'ω-mot périodique u^ω a uniquement une L-factorisation. On note $\mathcal{L}_n$ la classe des langages n-réduits.

Par définition, un langage $(n+1)$-réduit est n-réduit. On dit simplement que L est un *langage réduit* si L est 1-réduit.

D'après la proposition 1.4.6, un code est ∞-réduit.

Exemple 14 (reprise de p. 17). Le langage $L = \{a, ab, ba\}$ n'est pas réduit car l'ω-mot $(ab)^\omega$ a deux L-factorisations distinctes : $(ab, ab, \dots)$ et $(a, ba, ba, \dots)$.

Exemple 33. Soient $n > 1$ un entier et $A = \{a_0, a_1, \dots, a_n\}$ un alphabet. Le langage

$$L_n = \{a_0, a_0a_1, a_1a_2, \dots, a_{n-1}a_n, a_n\}$$

est $(n-1)$-réduit mais il n'est pas n-réduit car

$$(a_0)(a_1a_2)\dots(a_{n-1}a_n) = (a_0a_1)\dots(a_{n-2}a_{n-1})(a_n).$$

On peut coder l'alphabet $\{a_0, a_1, \dots, a_n\}$ par l'ω-code $\{a, ab, ab^2, \dots\}$ en posant $a_i \to ab^i$ pour obtenir

$$L'_n = \{a, ab^n\} \cup \{ab^iab^{i+1} : 0 \leq i \leq n\}$$

qui est $(n-1)$-réduit mais non n-réduit sur l'alphabet $\{a, b\}$.

Donc on a la relation suivante :

Proposition 5.1.2. $\mathcal{L}_1 \supsetneq \mathcal{L}_2 \supsetneq \cdots \supsetneq \mathcal{L}_\infty = Codes$

Le tableau qui suit donne le nombre maximal de L-factorisations d'un mot de type donné. L'astérisque $*$ indique que la propriété caractérise la classe de langages considérée.

Langage L	u^ω $(u \in L^{\leq n})$	u^ω $(u \in L^+)$	quelconque
ω-code	1	1	$1*$
code	1	$1*$	∞
langage n-réduit	$1*$	∞	∞

TABLE 5.1 – Comparaison du nombre maximal de factorisations sur les ω-codes, codes et langages n-réduits.

Maintenant, on peut prouver la caractérisation suivante pour les langages réduits :

Proposition 5.1.3. *Un langage $L \subseteq A^+$ est réduit si et seulement si pour chaque mot $u \in L$, on a $u^\omega \notin (L-u)^\omega$.*

Démonstration. La condition est clairement nécessaire. Réciproquement, supposons que L n'est pas réduit, c'est-à-dire il existe $u \in L$ tel que u^ω a deux L-factorisations de premiers pas différents : $(u, u, \dots)$ et $(v_0, v_1, \dots)$. Si $v_i \neq u$ pour tout entier $i \geq 0$, alors $u^\omega \in (L-u)^\omega$. S'il existe un plus petit entier $k > 0$ vérifiant $v_k = u$, alors on a

$$v_0v_1\dots v_{k-1}uv_{k+1}\cdots = uu^\omega = uv_0v_1\dots v_{k-1}uv_{k+1}\dots.$$

En comparant la longueur, on a donc

$$v_0v_1 \dots v_{k-1}u = uv_0v_1 \dots v_{k-1}.$$

On obtient $u^\omega = (v_0 \dots v_{k-1})^\omega$ et donc $u^\omega \in (L-u)^\omega$. □

Proposition 5.1.4. *Tout générateur réduit est un générateur minimal.*

Démonstration. Soit $L \subseteq A^+$ un langage. Si L n'est pas générateur minimal, alors il existe un mot $u \in L$ tel que $(L-u)^\omega = L^\omega$. Donc on a $u^\omega \in L^\omega = (L-u)^\omega$. D'après la proposition 5.1.3 donc L n'est pas réduit. □

Nous résumons ci-dessous les relations entre les différentes classes de générateurs que nous considérons :

Code générateur ⇒ Générateur réduit ⇒ Générateur minimal.

5.2 Décidabilité

L'objectif de cette section est de montrer que la propriété pour un langage d'être n-réduit est décidable sur les langages rationnels. D'abord, on rappelle des résultats de l'ensemble $Amb(L)$.

Soit $L \subseteq A^+$ un langage, $Amb(L)$ dénote l'ensemble des ω-mots de L^ω ayant plusieurs L-factorisations de premiers pas distincts [Kar85, JLP96], c'est-à-dire :

$$\begin{aligned} Amb(L) = \{w \in L^\omega : &\text{ il existe } (w_i)_{i\in\mathbb{N}}, (w_j{}')_{j\in\mathbb{N}} \\ &\text{deux } L\text{-factorisations de } w \text{ avec } w_0 \neq w_0{}'\} \end{aligned}$$

Remarque 5.2.1. Si chaque mot de L est représenté par une lettre d'un alphabet Σ comme on a fait dans le chapitre précédent. Ainsi on a la relation suivante entre $Amb(L)$ et $Amb_\Sigma(L)$:

$$\widetilde{Amb_\Sigma(L)} = L^* Amb(L).$$

Lemme 5.2.2. *[DLLS94] Si un langage $L \subseteq A^+$ est rationnel, alors l'ensemble $Amb(L)$ est rationnel.*

Démonstration. Si L est rationnel, la congruence qui est définie par $u \simeq v \Leftrightarrow u^{-1}L = v^{-1}L$ est d'index fini. On note $\langle u \rangle$ la classe d'équivalence du mot u. Donc l'ensemble $Amb(L)$ peut être calculé par $Amb(L) = \bigcup_{\langle u \rangle \subseteq L} \langle u \rangle (L^\omega \cap (u^{-1}L - \varepsilon)L^\omega)$. Notons que la famille des langages ω-rationnels est fermée par intersection finie. Ainsi $Amb(L)$ est rationnel. □

Lemme 5.2.3. *Soit n un entier positif. Un langage $L \subseteq A^+$ est n-réduit si et seulement si, pour tout $u \in L^{\leq n}$, on a $u^\omega \notin Amb(L)$.*

On a besoin aussi de quelques notations relatives à l'automate de Büchi et la congruence de Büchi [Büc60]. On rappelle que tout ω-langage rationnel est reconnaissable par un automate de Büchi complet [PP02].

Soit $\mathcal{A} = (A, Q, I, \delta, T)$ un automate de Büchi complet, pour chaque état $q \in Q$, et pour tout mot $u \in A^*$, on note

$$\delta_T(q, u) = \{q' \in Q : \text{il existe } t \in T \text{ et } \ u_1, u_2 \in A^* \\ \text{avec } u = u_1u_2 \text{ et } t \in \delta(q, u_1) \text{ et } q' \in \delta(t, u_2)\}$$

La congruence de Büchi $\approx$ est définie par

$$u \approx v \quad \Leftrightarrow \quad \forall q \in Q, \quad \begin{cases} \delta(q, u) = \delta(q, v) \\ \delta_T(q, u) = \delta_T(q, v) \end{cases}$$

pour tous mots $u, v \in A^+$. On note $[u] = \{w \in A^+ | w \approx u\}$ la classe d'équivalence du mot u. Comme $\approx$ est d'index fini calculable [Büc60], nous avons :

Lemme 5.2.4. *[LT87] Pour tout mot $u \in A^+$, sa classe d'équivalence $[u]$ est un langage rationnel constructible.*

Lemme 5.2.5. *Si $v_1 \approx v_2$, alors ${v_1}^\omega \in L_\omega(\mathcal{A}) \Leftrightarrow {v_2}^\omega \in L_\omega(\mathcal{A})$, où $L_\omega(\mathcal{A})$ dénote l'ω-langage reconnu par $\mathcal{A}$.*

On a la proposition suivante grâce aux lemmes précédents.

Proposition 5.2.6. *Soit n un entier positif. On peut décider si un langage rationnel est n-réduit.*

Démonstration. Soient L un langage rationnel et n un entier positif.

- On construit l'automate de Büchi qui reconnaît l'ensemble $Amb(L)$ (selon le lemme 5.2.2).
- On calcule les classes d'équivalences $[u_1], \ldots, [u_k]$ (selon le lemme 5.2.4).
- On vérifie s'il existe $[u_i]$ telle que

$$[u_i] \cap L^{\leq n} \neq \emptyset \text{ et } {u_i}^{\omega} \in Amb(L)$$

  Si oui alors L n'est pas n-réduit, sinon L est n-réduit (selon le lemme 5.2.3).

□

5.3 Langages réduits et codes générateurs

Dans la suite, on va présenter deux résultats sous l'hypothèse que *l'ω-langage rationnel L^{ω} possède un plus grand générateur dont la racine est un langage réduit.* Ces résultats sont des extensions de ceux (obtenus pour les codes) dans [Jul96].

Proposition 5.3.1. *Soit L un langage réduit tel que L^{+} est le plus grand générateur de L^{ω}. Tout générateur de L^{ω} contient une puissance de chaque élément de L.*

Démonstration. Comme L est réduit, pour chaque $u \in L$, u^{ω} a une seule factorisation $(u, u, \ldots)$ sur L. Soit G un générateur de L^{ω}. Comme $G \subseteq L^{+}$, chaque G-factorisation de u^{ω} est de la forme de :

$$(\underbrace{v_0}_{\in u^{+}}, \underbrace{v_1}_{\in u^{+}}, \ldots)$$

En conséquence, G contient une puissance de u. □

Proposition 5.3.2. *Soit L un langage réduit tel que L^{+} est le plus grand générateur de L^{ω}. L^{ω} a un générateur code préfixe si et seulement si L est un code préfixe.*

Démonstration. Soit L est un langage réduit mais non code préfixe tel que L^{+} est le plus grand générateur de L^{ω}. Alors il existe deux mots u et us dans L avec $s \neq \varepsilon$. Supposons au contraire que L^{ω} ait un générateur code préfixe P. D'après la proposition 5.3.1, alors il existe un entier $i \geq 1$ tel que $u^{i} \in P$. On considère

$u^i(su)^\omega = u^{i-1}(us)^\omega \in L^\omega = P^\omega$. Comme P est code préfixe on a $P^{-1}P^\omega = P^\omega = L^\omega$. Donc $(su)^\omega \in P^{-1}P^\omega = L^\omega$. Donc l'$\omega$-mot $(us)^\omega = u(su)^\omega$ a deux L-factorisations. En conséquence, L n'est pas réduit. □

Sous cette même hypothèse, nous caractérisons le fait qu'une ω-puissance est engendrée par un code pur ou non. Voici la définition d'un code pur [BPR09].

Soit $C \subseteq A^+$ un code, C est *pur* s'il vérifie la condition :

$$u^n \in C^+ \quad \text{implique} \quad u \in C^+$$

pour tout $u \in A^+$ et pour tout $n \geq 1$.

Proposition 5.3.3. *Soit R un langage réduit tel que R^+ est le plus grand générateur de R^ω. Si R n'est pas un code pur, alors il n'y a pas de générateur code pur de R^ω.*

Démonstration. Supposons qu'il existe un code $C \neq R$ tel que $C^\omega = R^\omega$. D'après le proposition 5.1.4, R est un générateur minimal, donc $C \not\subseteq R$. D'après la proposition 5.3.1, pour tout élément u de $R - C$, il existe un entier $i > 1$ tel que $u^i \in C$. Ainsi, C est non pur car $u^i \in C$ bien que $u \notin C^+$. □

5.4 Générateurs réduits

On va démontrer qu'il existe une ω-puissance qui ne peut pas être engendrée par un langage réduit. Cet exemple a déjà été considéré dans la section 2.2, où on a montré qu'il n'y a pas de code générateur.

Exemple 13 (reprise de p. 16). Soit $L = \{a^2, a^3, ba, b\}$. On rappelle que $L^\omega = a^2\{a, b\}^\omega$. Supposons qu'il existe R générateur réduit de L^ω. Tout d'abord, on montre que :

Fait 5.4.1. *Pour tout $k \geq 1$ et $u \in A^*$, on a $|a^k u a^* \cap R| \leq 1$.*

Démonstration du Fait. S'il existe $i > j \geq 0$ tels que $\{a^k u a^i, a^k u a^j\} \subseteq R$, alors

$$(a^k u a^i)^\omega = (a^k u a^j)\underbrace{(a^{i-j} a^k u a^i)^\omega}_{\in R^\omega}.$$

En conséquent, $(a^k ua)^\omega$ a deux R-factorisations : $(\alpha_i)_{i\geq 0}$ et $(\beta_j)_{j\geq 0}$ avec $\alpha_0 = a^k ua^i$ et $\beta_0 = a^k ua^j$. Donc R n'est pas réduit. □

- comme $a^\omega \in L^\omega$, il existe un unique entier $i_0 > 1$ tel que $a^{i_0} \in R$ (selon le fait précédent avec $u \in a^*$).
- comme $a^{i_0}aba^\omega \in L^\omega$ et $aba^\omega \notin L^\omega$, il existe un unique entier $i_1 \geq 0$ tel que $a^{i_0}aba^{i_1} \in R$.
- comme $a^{i_0}aba^{i_1}aba^\omega \in L^\omega$ et $a^{i_0}aba^{i_1}a \notin W$ (selon le fait précédent), donc il existe un unique entier $i_2 \geq 0$ tel que $a^{i_0}aba^{i_1}aba^{i_2} \in R$.
- et ainsi de suite, on définit de façon unique la suite infinie $(i_j)_{j\geq 0}$.

Maintenant, on considère l'ω-mot

$$w = a^{i_0}aba^{i_1}ab \dots aba^{i_n} \dots,$$

ce mot w appartient bien à L^ω mais il n'a pas de factorisation sur R. Ainsi R n'est pas générateur de L^ω. On en déduit que L^ω n'a pas de générateur réduit.

En conclusion, on a :

Proposition 5.4.2. *Il existe des ω-puissances finiment engendrées qui n'ont pas de générateur réduit.*

Perspectives & questions ouvertes

Nous avons étudié dans la section 4.5 des exemples pour étendre les langages à un relateur. D'une part dans l'exemple $\{a, ab, ba\}$ on peut considérer que a et b représentent chacun un ensemble de lettres (ou de mots). D'autre part on peut envisager les langages à deux relateurs.

De plus, on trouve dans le chapitre 4 que la structure de l'ensemble des relateurs pour un langage à un relateur ne dépend que du type des chevauchements des mots dans le relateur de base. Par exemple, le graphe de domino du langage à un relateur $02 \cong 10$ est « assez similaire » à celui du langage à un relateur $032 \cong 103$. Ainsi nous pouvons encoder le relateur de base, peut-être en posant $03 = 0'$, on a le langage à un relateur $0'2 \cong 10'$.

Par ailleurs, comme l'étiquette des sommets des graphes de domino n'est pas importante pour notre étude, nous pouvons chercher une méthode qui permet de construire le graphe de domino d'un langage à partir des relateurs de base.

Une autre piste possible est l'étude d'ω-puissance où le plus grand générateur existe et est engendré par un langage à trois éléments. Actuellement on ne sait pas décider si un langage à trois éléments a la même puissance ω qu'un ω-code. Plusieurs exemples considérés dans le chapitre 4 sont des langages à trois éléments et il semble que notre méthode soit bien adaptée pour résoudre ce cas.

Enfin, il faut noter que la décidabilité de langages à un relateur n'est pas montrée. Comment décider si un langage fini (ou rationnel) est un langage à un relateur ?

Bibliographie

[Aug01] X. Augros, *Des algorithmes autour des codes rationnels*, Ph.D. thesis, Université de Nice, Sophia Antipolis, 2001. 27

[BC04] M.-P. Béal and O. Carton, *Determinization of transducers over infinite words : The general case*, Theory Comput. Syst. **37** (2004), no. 4, 483–502.

[BN80] L. Boasson and M. Nivat, *Adherences of languages*, Journal of Computer and System Sciences **20** (1980), 285–309. 4

[BO93] R.V. Book and F. Otto, *String-rewriting systems*, Texts and monographs in computer science, Springer, 1993.

[Bou06] N. Bourbaki, *Théorie des ensembles*, Éléments de mathématique, Springer, 2006.

[BPR09] J. Berstel, D. Perrin, and C. Reutenauer, *Codes and Automata*, Encyclopedia of Mathematics and its Applications, vol. 129, Cambridge University Press, November 2009, (619 pp.). 1, 5, 7, 14, 80

[Bru91] V. Bruyère, *Research topics in the theory of codes*, Asmics workshop Codes and Automata (1991). 14

[Büc60] J.R. Büchi, *On a decision method in restricted second order arithmetics*, International Congress on Logic, Methodology and Philosophy of Science, Stanford University Press, 1960, pp. 1–11. 78

[Car08] O. Carton, *Langages formels, calculabilité et complexité*, Vuibert, 2008.

[CHK97] C. Choffrut, T. Harju, and J. Karhumäki, *A note on decidability questions on presentations of word semigroups*, Theor. Comput. Sci. **183** (1997), no. 1, 83–92.

[CK07] E. Czeizler and J. Karhumäki, *On non-periodic solutions of independent systems of word equations over three unknowns*, Int. J. Found. Comput. Sci. **18** (2007), no. 4, 873–897. 46

[Dev93] J. Devolder, *Codes, mots infinis et bi-infinis*, Ph.D. thesis, Université de Lille, 1993. 17

[DLLS94] J. Devolder, M. Latteux, I. Litovsky, and L. Staiger, *Codes and infinite words*, Acta Cybernetica **11** (1994), no. 4, 241–256. viii, 7, 8, 9, 77

[Eil74] S. Eilenberg, *Automata, languages, and machines*, vol. A, Academic Press, New York, 1974. 1

[GM59] E. N. Gilbert and E. F. Moore, *Variable length binary encodings*, Bell System Tech. J. **38** (1959), 933–967. 7

[Guz99] F. Guzmán, *Decidability of codes*, Journal of Pure and Applied Algebra (1999), 13–35. 27, 29, 30

[How95] J.M. Howie, *Fundamentals of semigroup theory*, Clarendon Press, Oxford, 1995. 36

[HW95] T. Head and A. Weber, *Deciding multiset decipherability*, IEEE Transactions on Information Theory **41** (1995), no. 1, 291–297. 27

[JLP96] S. Julia, I. Litovsky, and B. Patrou, *On codes, ω-codes and ω-generators*, Information Processing Letters **60** (1996), no. 1, 1–5. vii, 15, 77

[JT07] S. Julia and V.D. Tran, *Reduced languages as ω-generators*, Developments in Language Theory, 2007, pp. 266–277. 16

[JT09] ———, *Families and ω-ambiguity removal*, 7^{th} In. Conf. on Words (WORDS), 2009. 27

[Jul96] S. Julia, *On ω-generators and codes*, 23^d ICALP (Int. Coll. on Automata, Languages and Programming) (Berlin), Lecture Notes in Computer Sciences, vol. 1099, Springer, 1996, pp. 393–402. 79

[Jul97] ———, *A characteristic language for rational ω-power*, Developments in Language Theory, 1997, pp. 299–308. 6, 13, 16

[Kar85] J. Karhumäki, *On three-element codes*, Theoretical Computer Science **40** (1985), 3–11. 77

[Lal79] G. Lallement, *Semigroups and combinatorial applications*, Pure and applied mathematics, Wiley, 1979. 27

[Lit88] I. Litovsky, *Générateurs des langages rationnels de mots infinis*, Ph.D. thesis, Université de Lille, 1988. 13

[Lit89] ———, *Rank of rational finitely generated ω-languages*, FCT (Berlin), Lectures Notes in Computer Science, vol. 380, Springer, 1989, pp. 308–317. 13, 14

[Lit91] ———, *Prefix-free languages as ω-generators*, Information Processing Letters **37** (1991), 61–65. 15

[Lot02] M. Lothaire, *Algebraic combinatorics on words*, Encyclopedia of Mathematics and its Applications, vol. 90, Cambridge University Press, Cambridge, 2002. 2, 14

[LT86] M. Latteux and E. Timmerman, *Finitely generated ω-langages*, Information Processing Letters **23** (1986), 171–175. 14

[LT87] I. Litovsky and E. Timmerman, *On generators of rational ω-power languages*, Theoretical Computer Science **53** (1987), 187–200. 12, 47, 78

[McN98] R. McNaughton, *The finiteness of finitely presented monoids*, Theor. Comput. Sci. **204** (1998), no. 1-2, 169–182.

[McN01] ———, *Semi-thue systems with an inhibitor*, J. Autom. Reasoning **26** (2001), no. 4, 409–431.

[Niv81] M. Nivat, *Infinitary relations*, CAAP '81 : Proceedings of the 6th Colloquium on Tree in Algebra and Programming, Springer, 1981, pp. 46–75. 36

[PP02] D. Perrin and J. E. Pin, *Infinite words*, Academic Press, 2002. 1, 4, 78

[Pri01] C. Prieur, *How to decide continuity of rational functions on infinite words*, Theor. Comput. Sci. **250** (2001), no. 1-2, 71–82. 36

[Sén96] G. Sénizergues, *On the termination problem for one-rule semi-thue system*, RTA, 1996, pp. 302–316.

[SP53] A.A. Sardinas and C.W. Patterson, *A necessary sufficient condition for the unique decomposition of coded messages*, IRE Internat. Conv. Rec. (1953), no. 8, 104–108. 14

[Spe75] J.C. Spehner, *Quelques constructions et algorithmes relatifs aux sous-monoïdes d'un monoïde libre*, Semigroup Forum (1975). 27

[Sta80] L. Staiger, *A note on connected ω-langages*, Elektronische Informationsverarbeitung und Kybernetik **16** (1980), no. 5/6, 245–251. 4

[Sta86] ———, *On infinitary finite length codes*, Theoretical Informatics and Applications **20** (1986), no. 4, 483–494. vii, 1, 6, 7, 15

[Tho90] W. Thomas, *Automata on infinite objects*, Handbook of Theoretical Computer Science, vol. B, Elsevier Science Publishers, 1990. 1

[TL10] V.D. Tran and I. Litovsky, *One-relation languages and ω-code generators*, Proc. of the 13^{th} Mons Theoretical Computer Science Days, 2010.

[Wra90] C. Wrathall, *Confluence of one-rule thue systems*, IWWERT, 1990, pp. 237–246.

Résumé

Le sujet de cette thèse est l'étude des langages de mots infinis, en particulier les puissances infinies de langages de mots finis (puissance ω). Plus précisément, nous nous intéressons à la question ouverte suivante : étant donné un langage L, existe-t-il un ω-code C tel que $C^\omega = L^\omega$? Cette question est l'analogue de celle pour la concaténation finie : un sous-monoïde d'un monoïde libre est-il engendré par un code ou non ?

Dans un premier temps, nous étudions l'ensemble des relateurs d'un langage L, c'est-à-dire les couples de factorisations différentes d'un même mot de $L^* \cup L^\omega$; nous établissons une condition nécessaire pour que L^ω ait un code ou un ω-code générateur. Ensuite, nous définissons une nouvelle classe de langages : les langages à un relateur. Leurs ensemble de relateurs est le plus simple possible sans qu'ils soient des codes. Pour cette classe intéressante de langages, on caractérise les langages L tels qu'il existe un ω-code ou un code C tels que $L^\omega = C^\omega$. On montre que C ne peut pas être un langage fini. Enfin, une caractérisation des codes concernant les mots infinis nous amène à définir les langages réduits ; nous considérons les propriétés de ces langages en tant que générateurs de langages de mots infinis.

Mots-clés : Code, Combinatoire de mot, Langage formel, Générateur, Mot infini, ω-code, ω-puissance.

Abstract

This thesis deals with the languages of infinite words which are the ω-powers of a language of finite words. In particular, we focus on the open question : given a language L, does there exist an ω-code C such that $C^\omega = L^\omega$? It is quite similar to the question deciding whether a submonoid of a free monoid is generated by a code.

First, we study the set of relations satisfied by language L, i.e. the double factorizations of a word in $L^* \cup L^\omega$. We establish a necessary condition for that L^ω has a code or an ω-code generator. Next, we define the new class of languages where the set of relations is as simple as possible after codes : one-relation languages. For this class of languages, we characterize the languages L such that there exists a code or an ω-code C such that $L^\omega = C^\omega$, and we show that C is never a finite language. Finally, a characterization of codes concerning infinite words leads us to define reduced languages. We consider the properties of these languages as generators of languages of infinite words.

Keyworks : Code, Combinatoric on words, Formal language, Generator, Infinite word, ω-code, ω-power.